Lecture 13:

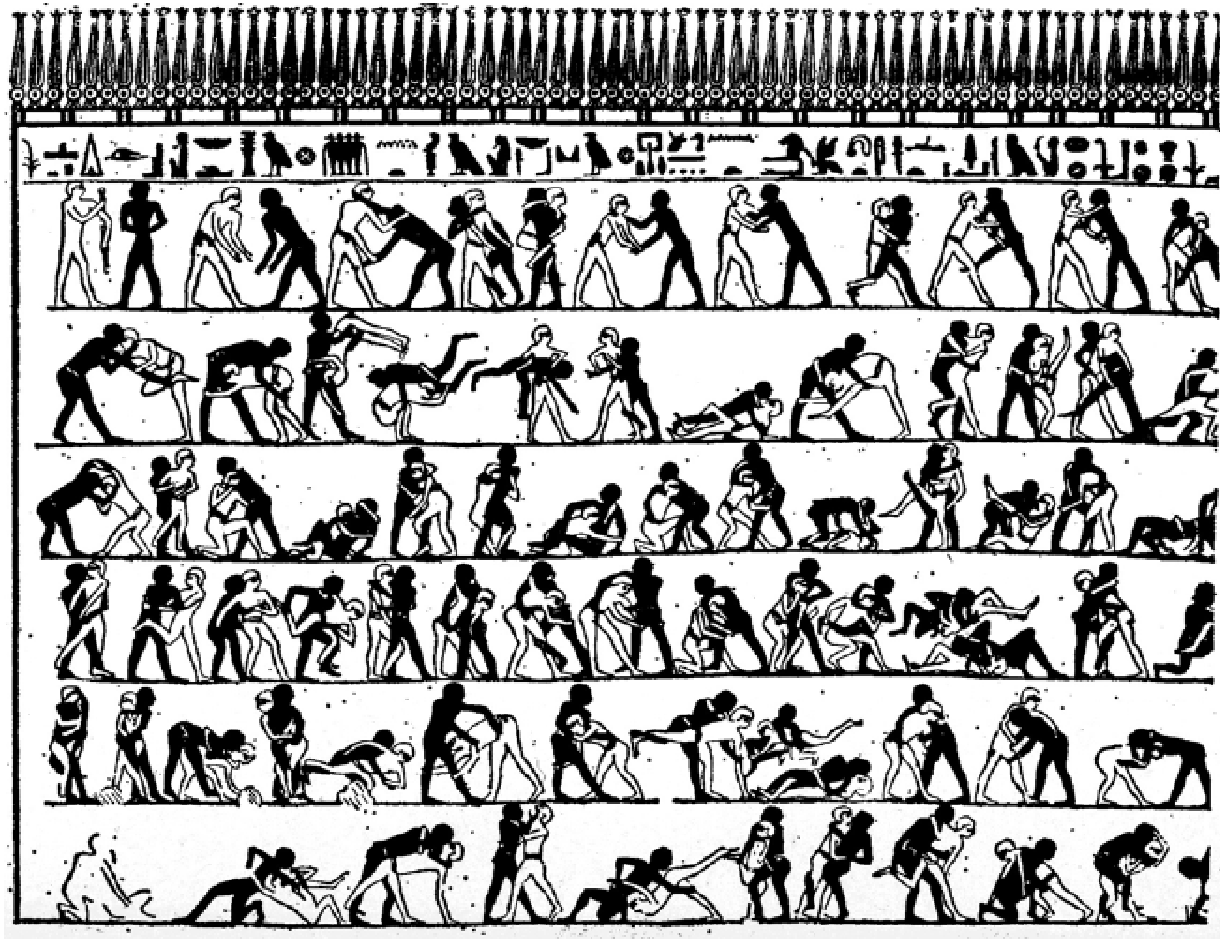
Animation

Contents

Before Animation

Shahr-e Sukhteh, Iran 3200 BCE

Before Animation



Tomb of Khnumhotep, Egypt 2400 BCE

The Phenakistoscope

- First systematic truly moving animation - the *phenakistoscope* (to be viewed in the mirror through the slit in the spinning disc)

PHENAKISTOSCOPE - Tribute to Joseph Plateau - - YouTube

First Film

- Used for research purposes
  in order to answer the question: *do horses life all four limbs off the ground in gallop?*

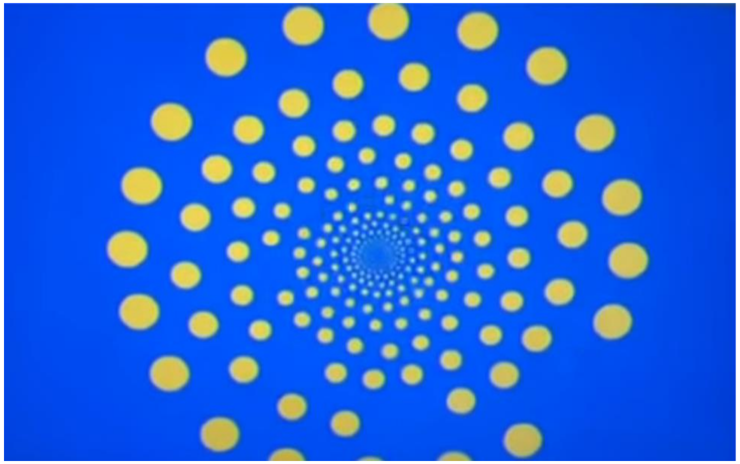
[Sallie Gardner at a Gallop (1878) - YouTube]

Computer animation is a sequence of still images rapidly changing at a fixed rate

The mechanism:

- Retinal persistence (our light receptors hold the perceived state over a couple of milliseconds) scientifically disproved
- **Beta phenomenon**: visual memory in brain - not eyeball
- **Phi phenomenon**: brain anticipates, giving sense of motion (it's Gestalt psychology again!)

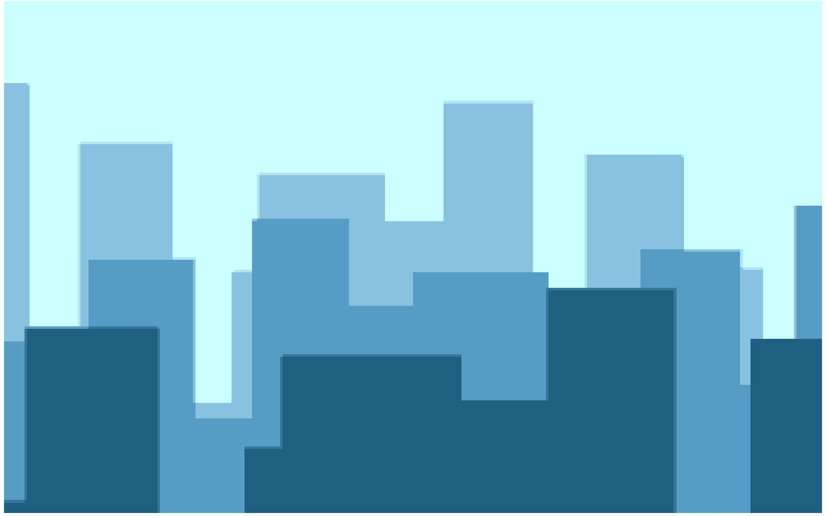
Animation basics: The optical illusion of motion - TED-Ed - YouTube



Phi Phenomenon - YouTube

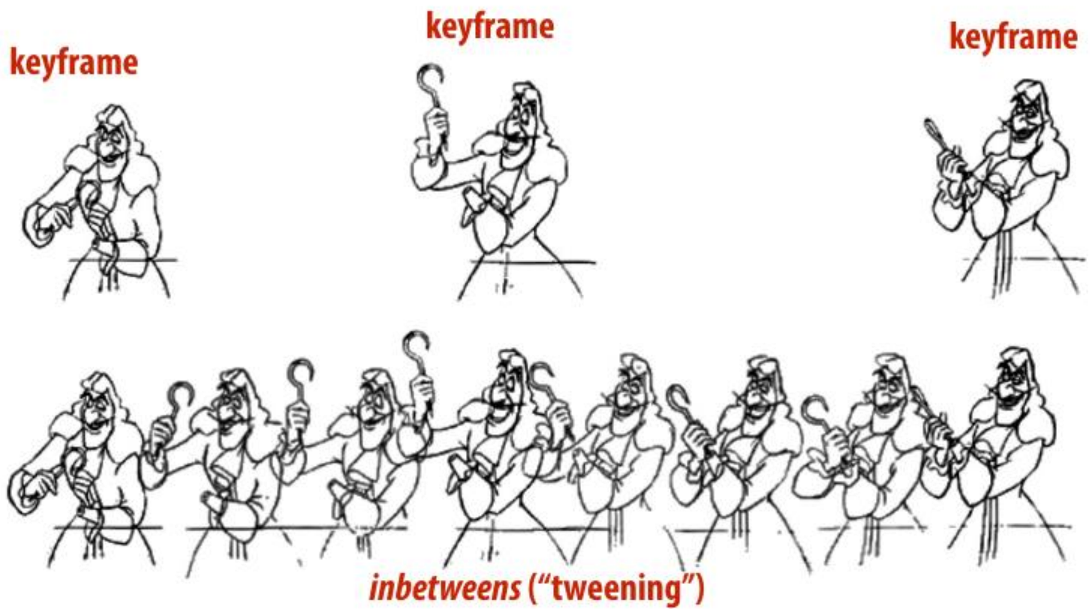
Motion

- Motion is a pre-attentive phenomenon
  - → It has a stronger power to render things distinguishable for us than color, shape, ...
- Back to Human Visual System: our eyes are more sensitive to motion at periphery
  - That's why we are prone to see "ghosts" in the corner of our visual field
- Motion triggers the orienting response / reflex (an organism's immediate response to a change in its environment, when that change is not sudden enough to elicit the startle reflex)
- Motion parallax provide 3-D cue (like stereopsis) – it means that we can understand depth in moving scenes despite not having the stereo-visual observation



“The Disney workflow”

- Senior artist draws keyframes
- Assistant draws in-betweens (tedious and labor intensive process)



In modern animation software the workflow is similar

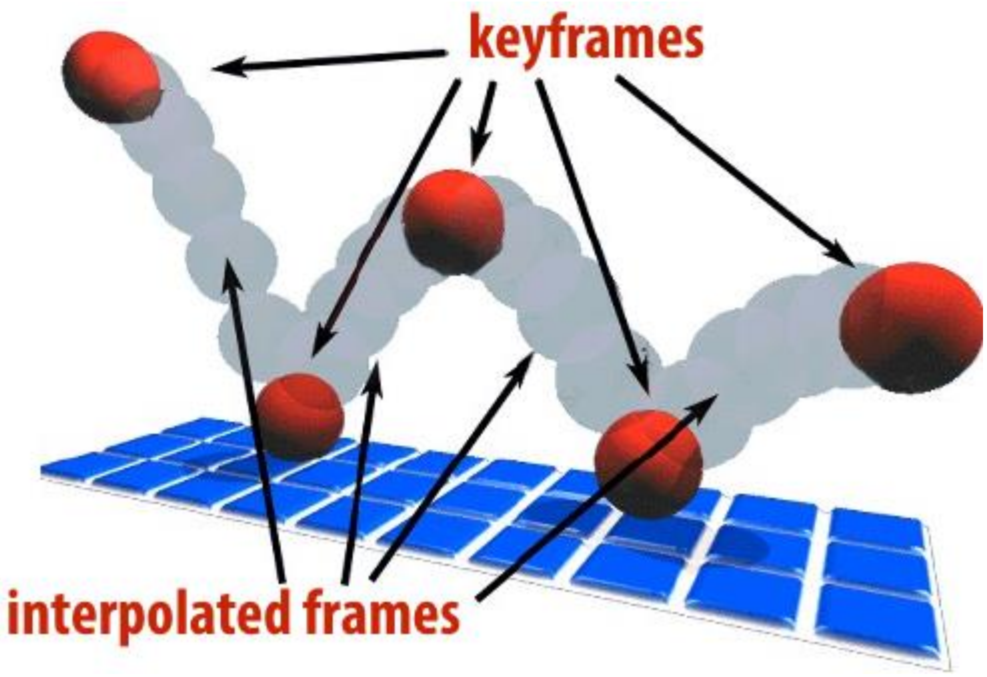
- You, as an artist decide on the key moments of the movement, and the software interpolates the geometry in the timesteps in between

Basic idea:

- Specify important events only
- Fills in the rest via interpolation / approximation

Key frames / Events:

- Position
- Color
- Light intensity
- Camera zoom
- *etc.*



Camera

- Position
- Direction
- Focal length

Light Source

- Position
- Direction
- Radiant Power

Geometry

- Position
- Affine Transform
  - Rotation
  - Motion
  - Scaling
  - Shearing

Shading

- Transparency
- Textures
- Diffuse properties
- *etc.*

Example

- **Position** is one of the most common characteristics, which is provided via `Vec3f` values
- If the sequence contains 240 frames, for object **A** we can assign *e.g.* frames 0, 100 and 240 as *keyframes* and for object **B** - frames 10, 20 and 200
- Next we need to provide 3 positions for object **A** and 3 positions for object **B** for every keyframe, *e.g.*
  - ```
    A.pos1 = Vec3f(7, 0, 1);
    A.pos2 = Vec3f(10, 0, 10);
    ```
- For the frames lying in-between 0 and 100, interpolate the position of object **A** using `A.pos1` and `A.pos2`
- By analogy proceed with object **B** and all other frames
    ```

Curve descriptions

- Explicit:
  - $y = f(x)$
    - $y(x) = \pm\sqrt{r^2 - x^2}$ restricted domain
- Implicit:
  - $F(x, y) = 0$
    - $x^2 + y^2 - r^2 = 0$ unknown solution set
- Parametric:
  - $\mathrm{x} = f_x(t),\ y = f_y(t)$
    - $\begin{matrix} x(t) = r\cos 2\pi t \\ y(t) = r\sin 2\pi t \end{matrix}, \quad t \in [0, 1]$ flexibility and ease of use

Polynomials

- $x(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3 + \cdots$
- Avoids complicated functions (*e.g.* $pow(), \exp(), \sin(), sqrt()$)
- Use simple polynomials of low degree

Monomial basis

- Simple basis: $1, t, t^2, \dots$ (t usually in $[0, 1]$)

Polynomial representation

$$
\begin{aligned}
x(t) &= a_0 + a_1 t + a_2 t^2 + \boxed{a_3} t^3 + \cdots \\
y(t) &= b_0 + b_1 t + b_2 t^2 + \boxed{b_3} t^3 + \cdots \\
z(t) &= c_0 + c_1 t + \boxed{c_2 t^2} + \boxed{c_3} t^3 + \cdots
\end{aligned}
$$

Degree

Coefficients $p_i \in \mathbb{R}^3$

Monomials

$$
P(t) = \begin{pmatrix} x(t) \\ y(t) \\ z(t) \end{pmatrix} = \sum_{i=0}^{n} \begin{pmatrix} a_i \\ b_i \\ c_i \end{pmatrix} t^i
$$

- Coefficients can be determined from a sufficient number of constraints (*e.g.* interpolation of given points)
- Given $(n + 1)$ parameter values t_i and points P_i
- Solution of a linear system in the A_i - possible, but inconvenient

Matrix representation

$$
P(t)^\top = \begin{pmatrix} t^n & t^{n-1} & \cdots & t & 1 \end{pmatrix} \begin{pmatrix} a_n & b_n & c_n \\ a_{n-1} & b_{n-1} & c_{n-1} \\ \vdots & \vdots & \vdots \\ a_0 & b_0 & c_0 \end{pmatrix}
$$

Derivative

- Polynomial of degree $(n-1)$

$$\frac{dP(t)}{dt} = P'(t) = \begin{pmatrix} nt^{n-1} & (n-1)t^{n-2} & \cdots & 1 & 0 \end{pmatrix} \begin{pmatrix} a_n & b_n & c_n \\ a_{n-1} & b_{n-1} & c_{n-1} \\ \vdots & \vdots & \vdots \\ a_0 & b_0 & c_0 \end{pmatrix}$$

- Derivative at a point is equal to the tangent vector at that point

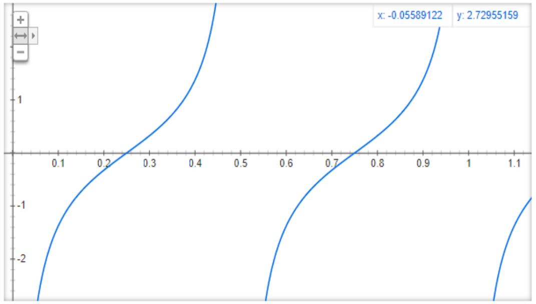
Example

$$P(t) = \begin{pmatrix} \cos 2\pi t & \sin 2\pi t \end{pmatrix} \begin{pmatrix} r & 0 \\ 0 & r \end{pmatrix}$$

$$P'(t) = \begin{pmatrix} -2\pi \cdot \sin 2\pi t & 2\pi \cdot \cos 2\pi t \end{pmatrix} \begin{pmatrix} r & 0 \\ 0 & r \end{pmatrix}$$

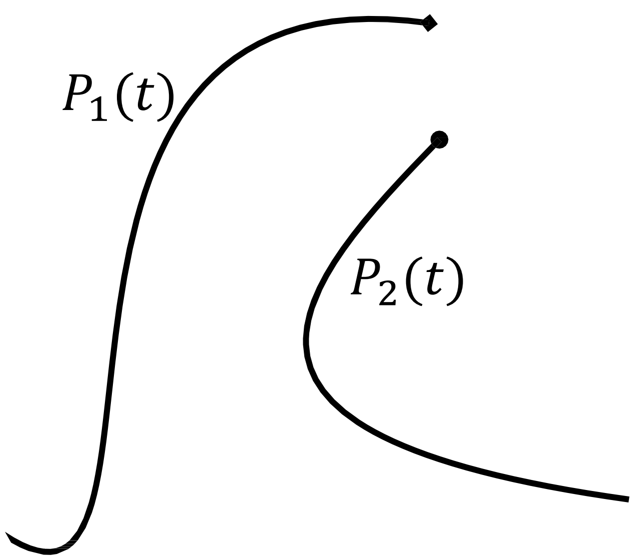
$$x'(t) = -2\pi r \cdot \sin 2\pi t$$
$$y'(t) = 2\pi r \cdot \cos 2\pi t$$

$$\frac{dy}{dx} = \frac{y'(t)}{x'(t)} = \frac{2\pi r \cdot \cos 2\pi t}{-2\pi r \cdot \sin 2\pi t} = -\operatorname{ctg} 2\pi t$$

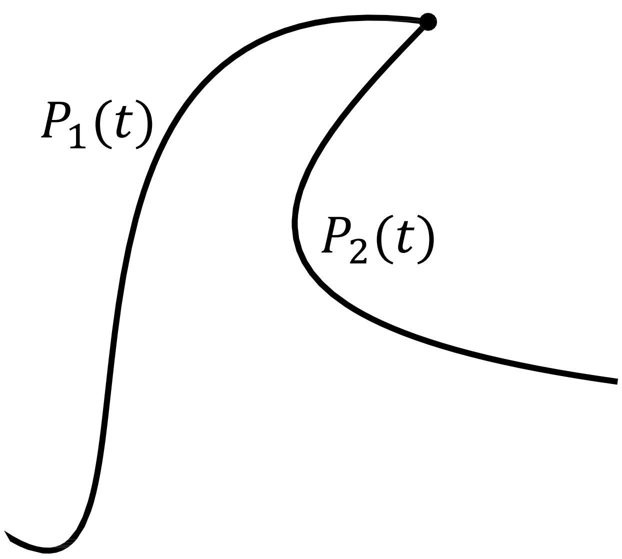


Continuity and smoothness between parametric curves

- There are two criteria for continuity:
  - Geometric continuity G^0
  - Parametric continuityC^0
- If curve P_1ends in the same point where curve P_2 starts, it is said that we have both G^0 and C^0continuity



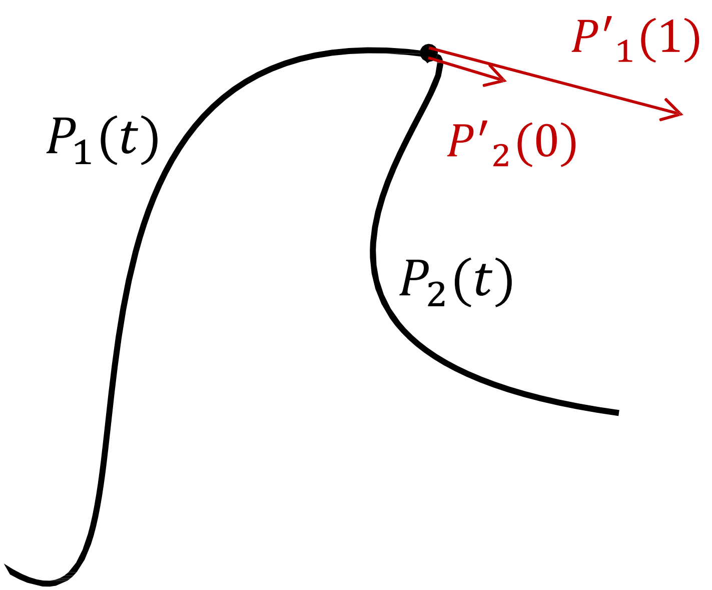
Not continuous
$P_1(1) \neq P_2(t)$



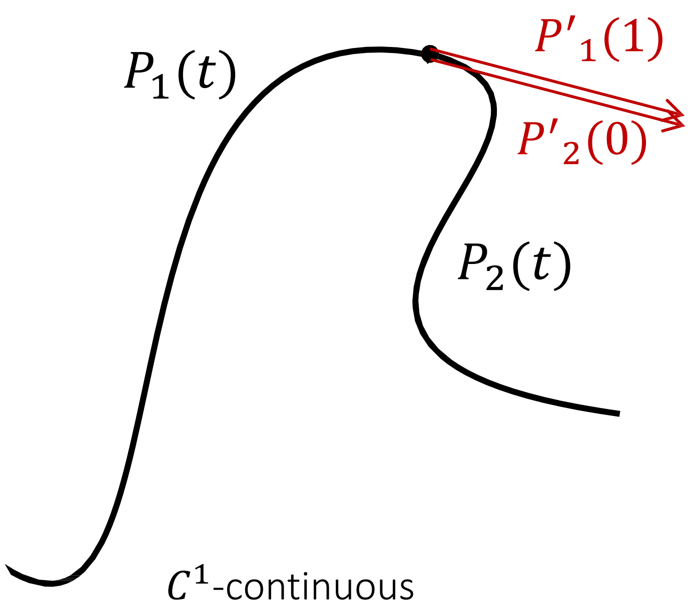
Continuous
$P_1(1) = P_2(0)$

Continuity and smoothness between parametric curves

- If the tangent vectors at the joint are equally directed $P'_1(1) = kP'_2(0)$
  - It is said that we have geometric continuity G^1
- If the tangent vectors at the joint are equal $P'_1(1) = P'_2(0)$
  - It is said that we have parametric continuity C^1
- Similar for higher derivatives



G^1 -continuous
G^0 + tangent vectors parallel
$P'_1(1) = kP'_2(0)$



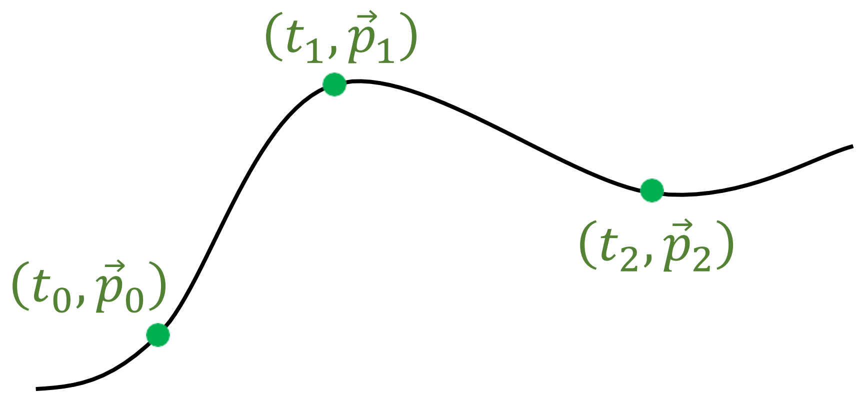
C^1-continuous
C^0 + tangent vectors parallel
$P'_1(1) = P'_2(0)$

Given a set of key-points:

- $(t_i, \vec{p}_i),\ t_i \in \mathbb{R},\ \ \vec{p}_i \in \mathbb{R}^d$

Find a polynomial P such that:

- $\forall i\ \ P(t_i) = \vec{p}_i$



Given a set of points:

- $(t_i, \vec{p}_i),\ t_i \in \mathbb{R},\ \ \vec{p}_i \in \mathbb{R}^d$

Find a polynomial P such that:

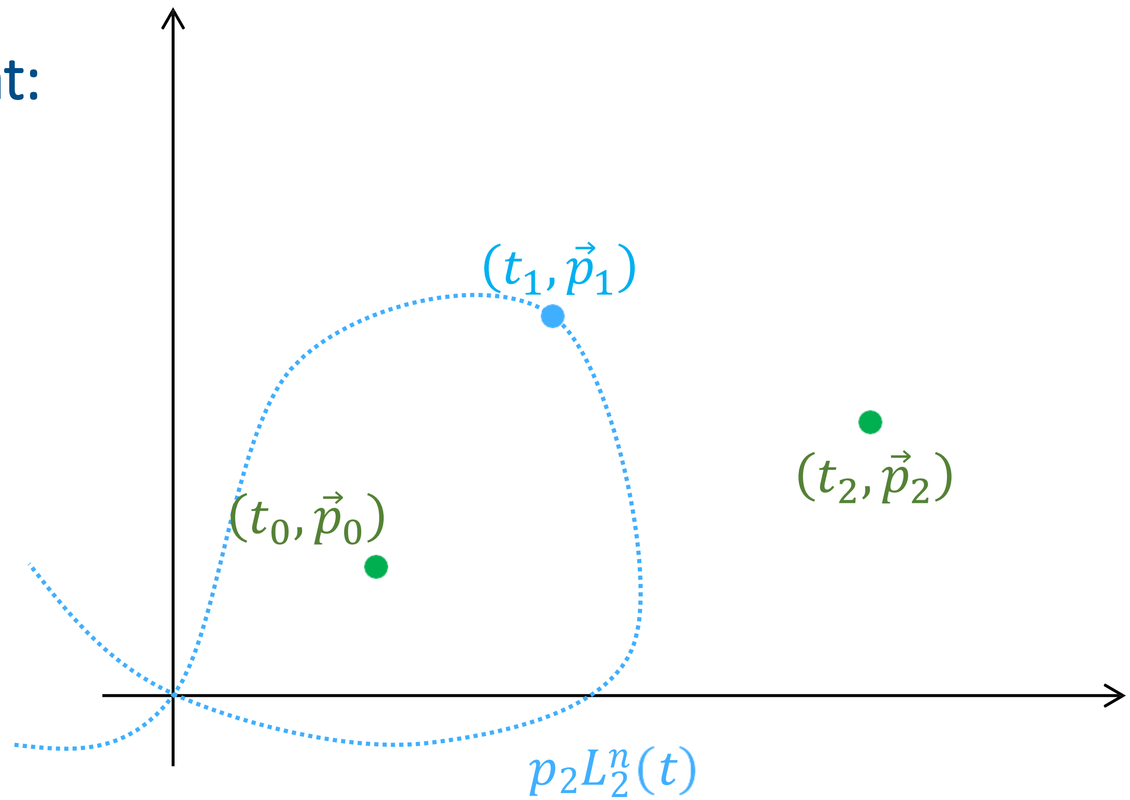
- $\forall i\ \ P(t_i) = \vec{p}_i$

For each point associate a *Lagrange basis polynomial:*

$$L_i^n(t) = \prod_{\substack{j=0 \\ i \neq j}}^{n} \frac{t - t_j}{t_i - t_j}$$

where

$$L_i^n(t_j) = \delta_{ij} = \begin{cases} 1 & i = j \\ 0 & otherwise \end{cases}$$



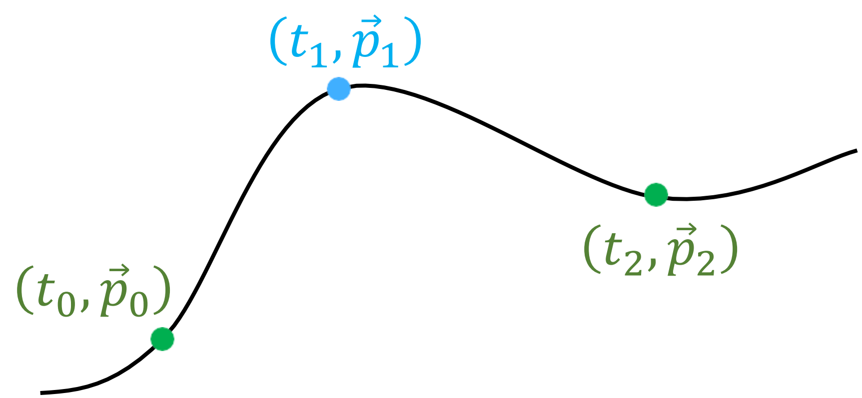
Given a set of points:

- $(t_i, \vec{p}_i),\ t_i \in \mathbb{R},\ \ \vec{p}_i \in \mathbb{R}^d$

Find a polynomial P such that:

- $\forall i\ \ P(t_i) = \vec{p}_i$

For each point associate a *Lagrange basis polynomial*:



$$L_i^n(t) = \prod_{\substack{j=0 \\ i \neq j}}^{n} \frac{t - t_j}{t_i - t_j}$$

where

$$L_i^n(t_j) = \delta_{ij} = \begin{cases} 1 & i = j \\ 0 & otherwise \end{cases}$$

Add the Lagrange basis with points as weights:

$$P(t) = \sum_{i=0}^{n} L_i^n(t)\vec{p}_i$$

$$P(t)^\intercal = \begin{pmatrix} L_0^n & L_1^n & \cdots & L_{n-1}^n \end{pmatrix} \begin{pmatrix} p_{0,x} & p_{0,y} & p_{0,z} \\ p_{1,x} & p_{1,y} & p_{1,z} \\ \vdots & \vdots & \vdots \\ p_{n-1,x} & p_{n-1,y} & p_{n-1,z} \end{pmatrix}$$

For each point associate a Lagrange basis polynomial:

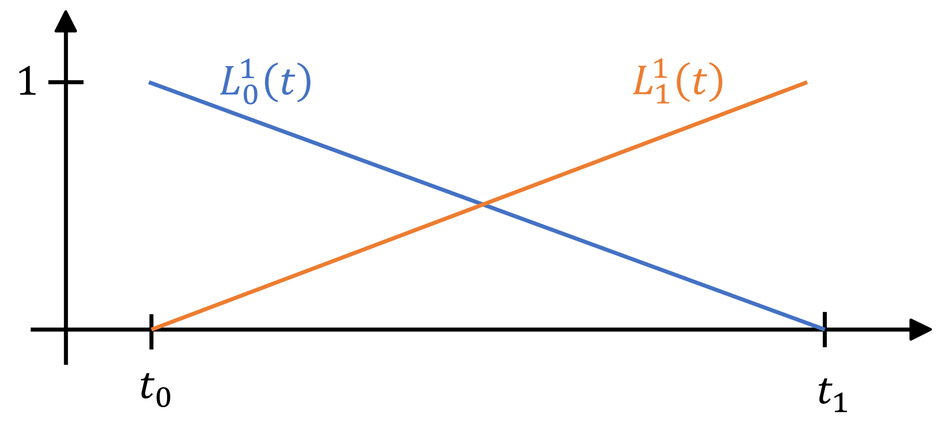
$$L_i^n(t) = \prod_{\substack{j=0 \\ i \neq j}}^{n} \frac{t - t_j}{t_i - t_j}$$

Simple Linear Interpolation

- $T = \{t_0, t_1\}$

$$L_0^1(t) = \frac{t - t_1}{t_0 - t_1}$$

$$L_1^1(t) = \frac{t - t_0}{t_1 - t_0}$$



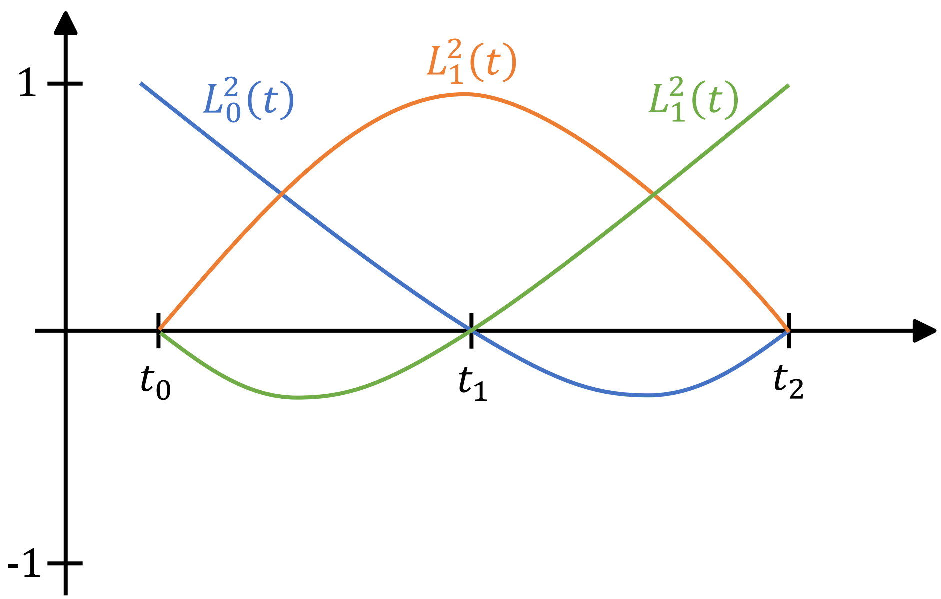
Simple Quadratic Interpolation

- $T = \{t_0, t_1, t_2\}$

$$L_0^2(t) = \frac{t - t_1}{t_0 - t_1} \frac{t - t_2}{t_0 - t_2}$$

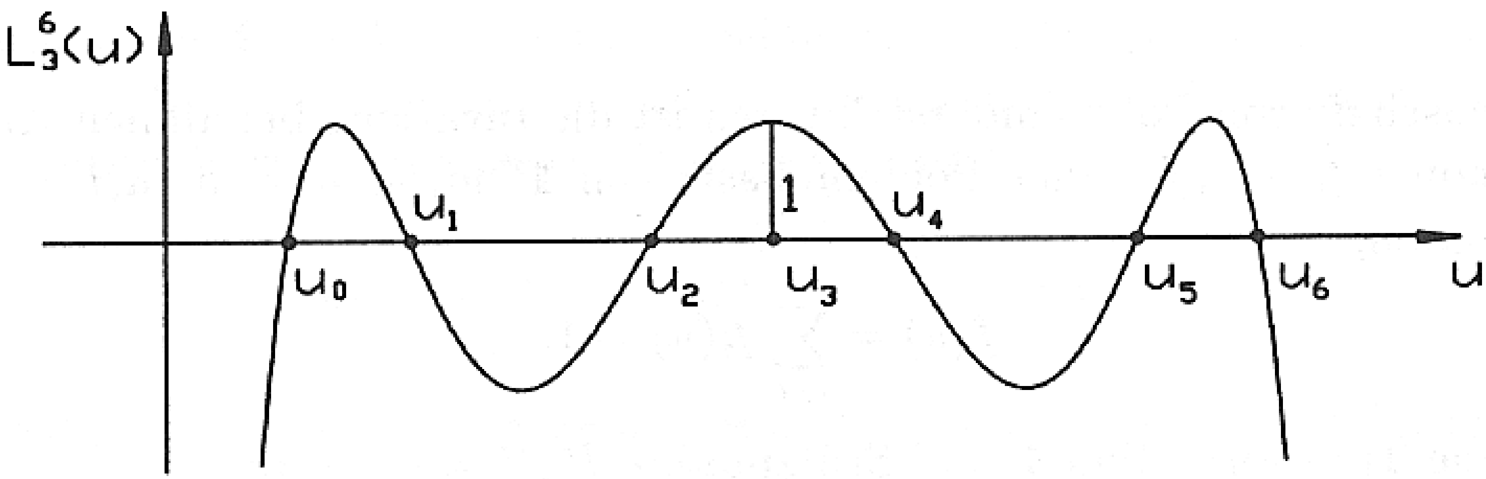
$$L_1^2(t) = \frac{t - t_0}{t_1 - t_0} \frac{t - t_2}{t_1 - t_2}$$

$$L_2^2(t) = \frac{t - t_0}{t_2 - t_0} \frac{t - t_1}{t_2 - t_1}$$



Problems with a single polynomial

- Degree depends on the number of interpolation constraints
- Strong overshooting for high degree ($n > 7$)
- Problems with smooth joints
- Numerically unstable
- No local changes

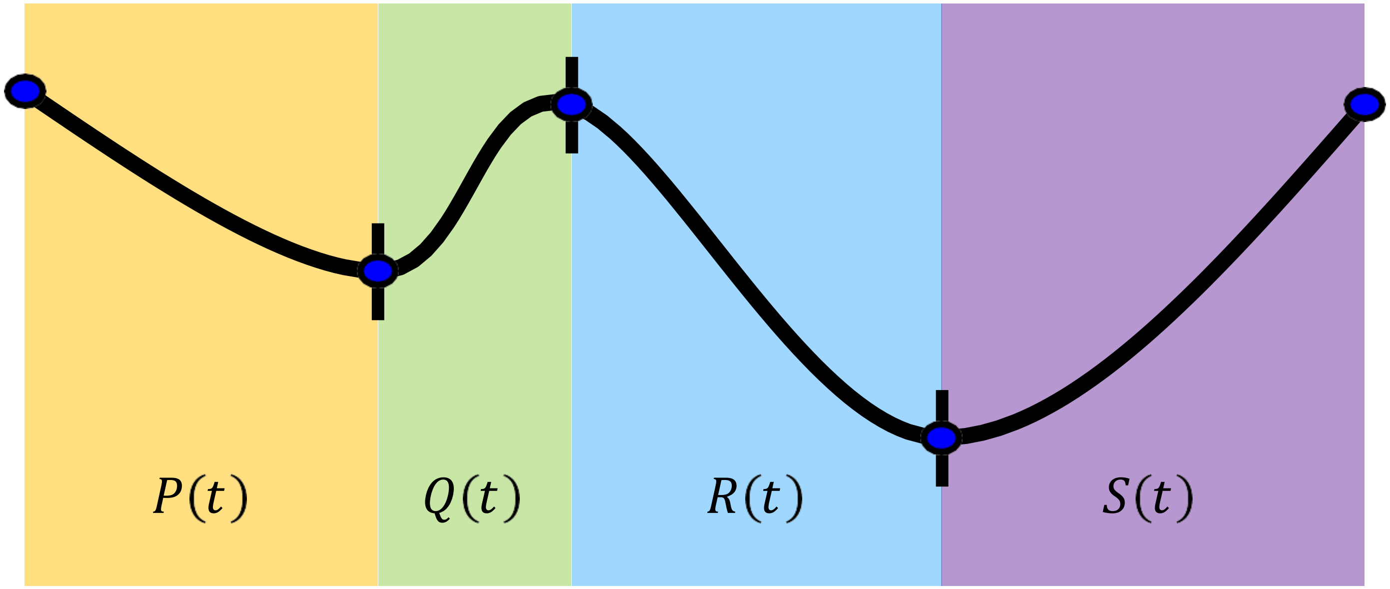


Functions for interpolation & approximation

- Standard curve and surface primitives in geometric modeling
- Key frame and in-betweens in animations
- Filtering and reconstruction of images

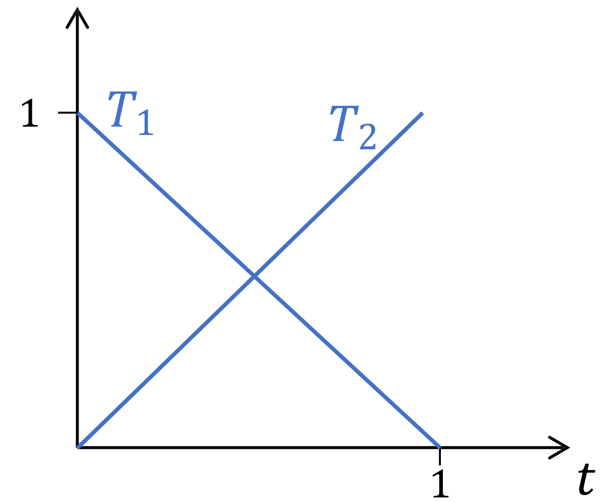
Historically

- Name for a tool in ship building
  - Flexible metal strip that tries to stay straight
- Within computer graphics:
  - Piecewise polynomial function



Linear splines

- Defined by two points: $\vec{p}_1, \vec{p}_2$
- Searching for $P(t)$ such that:
  - $P(0) = \vec{p}_1$
  - $P(1) = \vec{p}_2$
  - Degree of P is 1
- Basis:
  - $T_1(t) = 1 - t$
  - $T_2(t) = t$



$$P(t) = \vec{p}_1 T_1(t) + \vec{p}_2 T_2(t)$$

$$P(t)^\intercal = \underbrace{\boxed{(1 - t \quad t)}}_{\text{Linear basis}} \begin{pmatrix} \vec{p}_1^\intercal \\ \vec{p}_2^\intercal \end{pmatrix}$$

Linear Interpolation

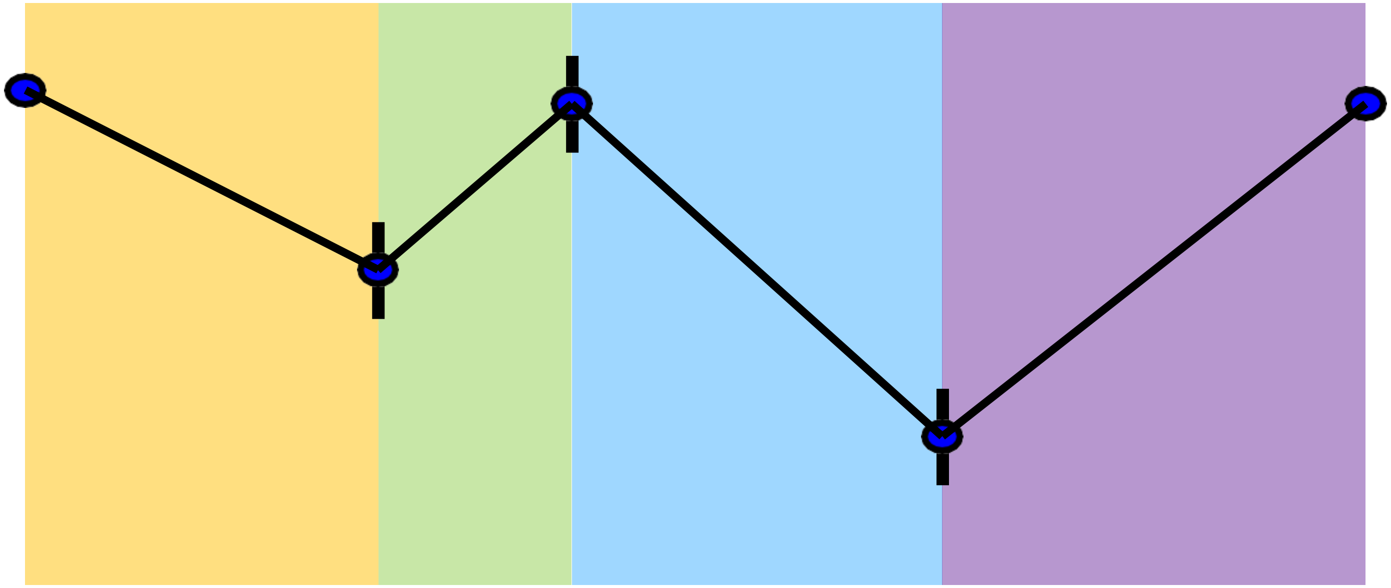
monomial basis M Linear basis

$$
\begin{tikzcd}
\begin{pmatrix} t & 1 \end{pmatrix} \arrow[rr, bend left, "\begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix}"] & & \begin{pmatrix} 1-t & t \end{pmatrix} \arrow[ll, bend left, "\begin{pmatrix} -1 & 1 \\ 1 & 0 \end{pmatrix}"]
\end{tikzcd}
$$

$$P(t)^{\intercal} = M \cdot \begin{pmatrix} -1 & 1 \\ 1 & 0 \end{pmatrix} \cdot \begin{pmatrix} \vec{p}_1^{\intercal} \\ \vec{p}_2^{\intercal} \end{pmatrix}$$

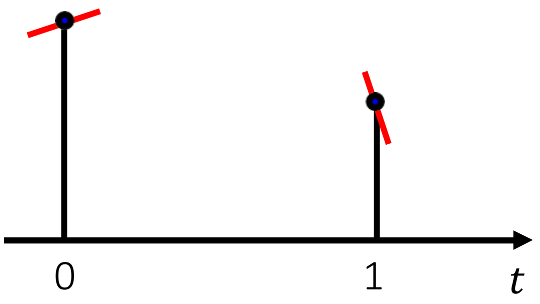
$$P(t)^{\intercal} = M \cdot \begin{pmatrix} -1 & 1 \\ 1 & 0 \end{pmatrix} \cdot \begin{pmatrix} \vec{p}_1^{\intercal} \\ \vec{p}_2^{\intercal} \end{pmatrix}$$

C^0-continuous



Cubic splines

- Defined by two points: $\vec{p}_1, \vec{p}_2$ and two tangents: $\vec{\tau}_1, \vec{\tau}_2$
- Searching for $P(t)$ such that:
  - $P(0) = \vec{p}_1$
  - $P'(0) = \vec{\tau}_1$
  - $P'(1) = \vec{\tau}_2$
  - $P(1) = \vec{p}_2$
  - Degree of P is 3
- Basis:
  - $H_0^3(t) = ?$
  - $H_1^3(t) = ?$
  - $H_2^3(t) = ?$
  - $H_3^3(t) = ?$



$$P(t) = P(0)H_0^3(t) + P'(0)\, H_1^3(t) + P'(1)\, H_2^3(t) + P(1)H_3^3(t)$$

Cubic splines

- Defined by two points: $\vec{p}_1, \vec{p}_2$ and two tangents: $\vec{\tau}_1, \vec{\tau}_2$
- Searching for $P(t)$ such that:
  - $P(0) = \vec{p}_1$
  - $P'(0) = \vec{\tau}_1$
  - $P'(1) = \vec{\tau}_2$
  - $P(1) = \vec{p}_2$
  - Degree of P is 3
- Basis:
  - $H_0^3(t) = ?$
  - $H_1^3(t) = ?$
  - $H_2^3(t) = ?$
  - $H_3^3(t) = ?$

$$
\begin{tikzcd}
\begin{pmatrix} t^3 & t^2 & t & 1 \end{pmatrix} \arrow[r, bend left, "H^{-1}"] & \begin{pmatrix} H_0^3(t) & H_1^3(t) & H_2^3(t) & H_3^3(t) \end{pmatrix} \arrow[l, bend left, "H"]
\end{tikzcd}
$$

$$
P(t)^\intercal = M \cdot H \cdot \begin{pmatrix} \vec{p}_1^\intercal \\ \vec{\tau}_1^\intercal \\ \vec{\tau}_2^\intercal \\ \vec{p}_2^\intercal \end{pmatrix} = M \cdot H \cdot G
$$

Cubic splines

- Defined by two points: $\vec{p}_1, \vec{p}_2$ and two tangents: $\vec{\tau}_1, \vec{\tau}_2$
- Searching for $P(t)$ such that:
  - $P(0) = \vec{p}_1$
  - $P'(0) = \vec{\tau}_1$
  - $P'(1) = \vec{\tau}_2$
  - $P(1) = \vec{p}_2$
  - Degree of P is 3
- Basis:
  - $H_0^3(t) = ?$
  - $H_1^3(t) = ?$
  - $H_2^3(t) = ?$
  - $H_3^3(t) = ?$

- $P(t)^\intercal = \begin{pmatrix} t^3 & t^2 & t & 1 \end{pmatrix} \cdot H \cdot G$
- $P'(t)^\intercal = \begin{pmatrix} 3t^2 & 2t & 1 & 0 \end{pmatrix} \cdot H \cdot G$

- $\vec{p}_1^\intercal = P(0)^\intercal = \begin{pmatrix} 0 & 0 & 0 & 1 \end{pmatrix} \cdot H \cdot G$
- $\vec{\tau}_1^\intercal = P'(0)^\intercal = \begin{pmatrix} 0 & 0 & 1 & 0 \end{pmatrix} \cdot H \cdot G$
- $\vec{\tau}_2^\intercal = P'(1)^\intercal = \begin{pmatrix} 3 & 2 & 1 & 0 \end{pmatrix} \cdot H \cdot G$
- $\vec{p}_2^\intercal = P(1)^\intercal = \begin{pmatrix} 1 & 1 & 1 & 1 \end{pmatrix} \cdot H \cdot G$

$$\begin{pmatrix} \vec{p}_1^\intercal \\ \vec{\tau}_1^\intercal \\ \vec{\tau}_2^\intercal \\ \vec{p}_2^\intercal \end{pmatrix} = \begin{pmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 3 & 2 & 1 & 0 \\ 1 & 1 & 1 & 1 \end{pmatrix} \cdot H \cdot \begin{pmatrix} \vec{p}_1^\intercal \\ \vec{\tau}_1^\intercal \\ \vec{\tau}_2^\intercal \\ \vec{p}_2^\intercal \end{pmatrix}$$

Cubic splines

- Defined by two points: $\vec{p}_1, \vec{p}_2$ and two tangents: $\vec{\tau}_1, \vec{\tau}_2$
- Searching for $P(t)$ such that:
    - $P(0) = \vec{p}_1$
    - $P'(0) = \vec{\tau}_1$
    - $P'(1) = \vec{\tau}_2$
    - $P(1) = \vec{p}_2$
    - Degree of P is 3
- Basis:
    - $H_0^3(t) =?$
    - $H_1^3(t) =?$
    - $H_2^3(t) =?$
    - $H_3^3(t) =?$

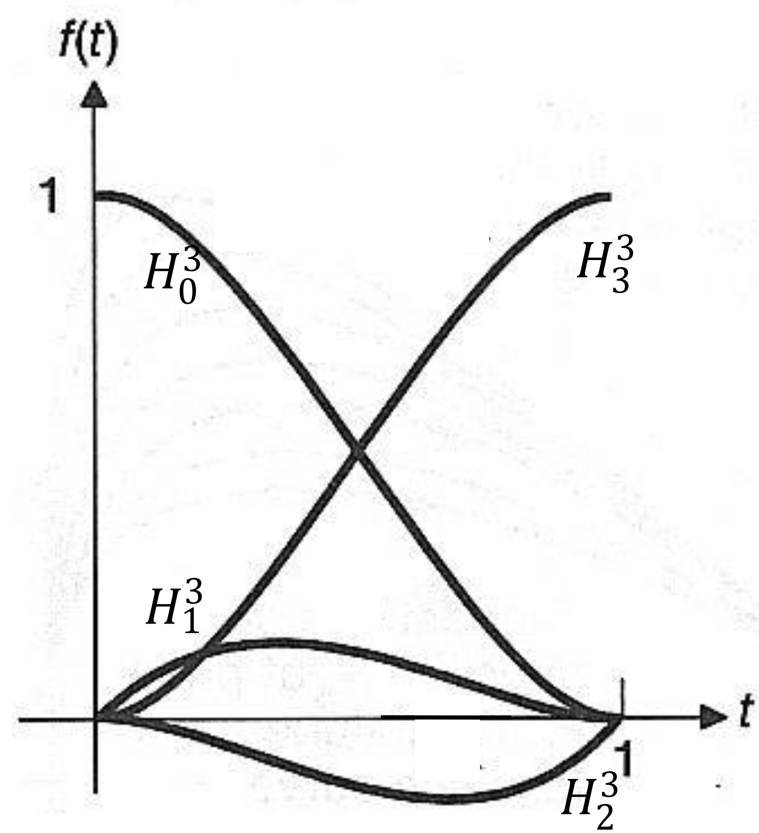
$$H = \begin{pmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 3 & 2 & 1 & 0 \\ 1 & 1 & 1 & 1 \end{pmatrix}^{-1} = \begin{pmatrix} 2 & 1 & 1 & -2 \\ -3 & -2 & -1 & 3 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix}$$

Cubic splines

- Defined by two points: $\vec{p}_1, \vec{p}_2$ and two tangents: $\vec{\tau}_1, \vec{\tau}_2$
- Searching for $P(t)$ such that:
  - $P(0) = \vec{p}_1$
  - $P'(0) = \vec{\tau}_1$
  - $P'(1) = \vec{\tau}_2$
  - $P(1) = \vec{p}_2$
  - Degree of P is 3
- Basis:
  - $H_0^3(t) = (1-t)^2(1+2t)$
  - $H_1^3(t) = t(1-t)^2$
  - $H_2^3(t) = t^2(t-1)$
  - $H_3^3(t) = (3-2t)t^2$

$$H = \begin{pmatrix} \boxed{\begin{matrix} 2 \\ -3 \\ 0 \\ 1 \end{matrix}} & \boxed{\begin{matrix} 1 \\ -2 \\ 1 \\ 0 \end{matrix}} & \boxed{\begin{matrix} 1 \\ -1 \\ 0 \\ 0 \end{matrix}} & \boxed{\begin{matrix} -2 \\ 3 \\ 0 \\ 0 \end{matrix}} \end{pmatrix}$$

$$(H_0^3(t) \quad H_1^3(t) \quad H_2^3(t) \quad H_3^3(t))$$



Cubic splines

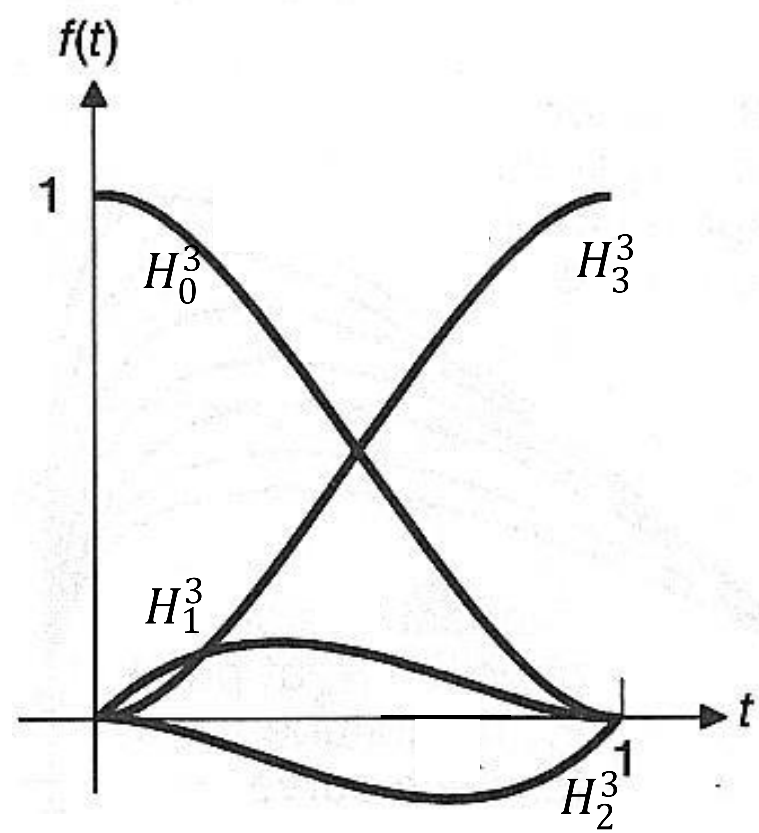
- Basis:
  - $H_0^3(t) = (1-t)^2(1+2t)$
  - $H_1^3(t) = t(1-t)^2$
  - $H_2^3(t) = t^2(t-1)$
  - $H_3^3(t) = (3-2t)t^2$

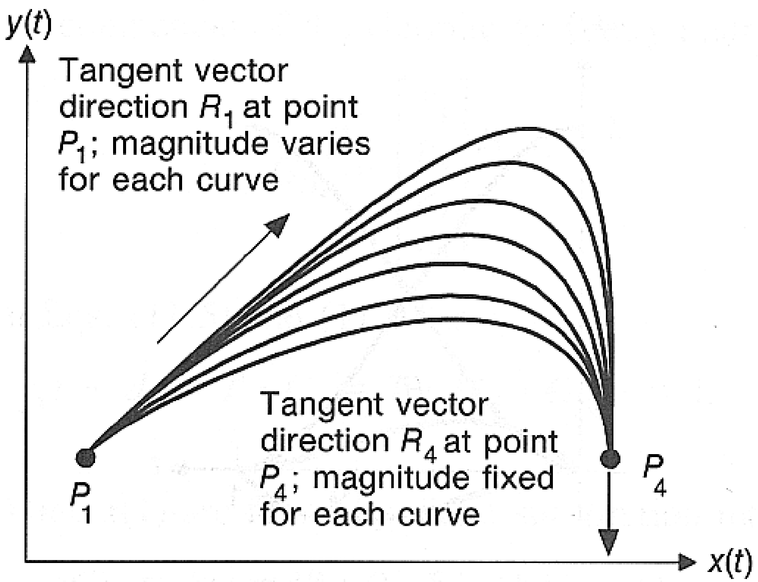
$$H = \begin{pmatrix} 2 & 1 & 1 & -2 \\ -3 & -2 & -1 & 3 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix}$$

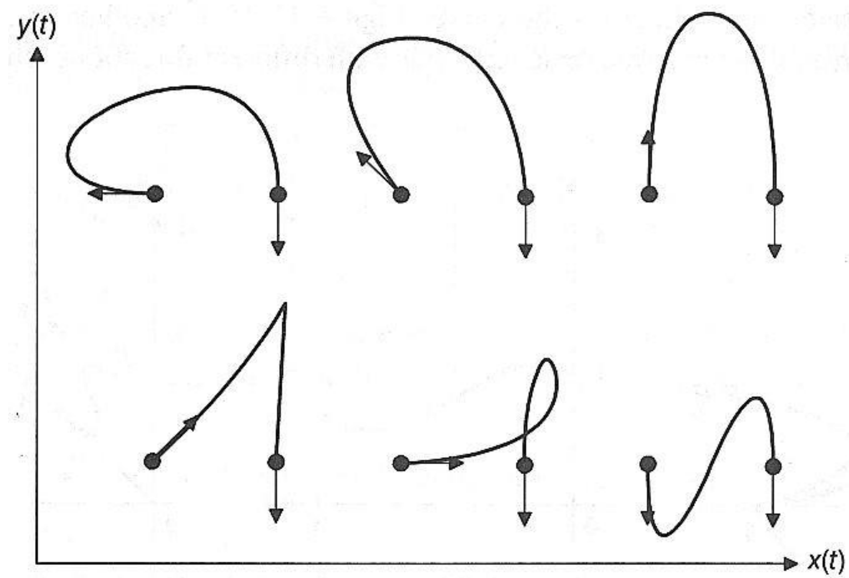
$$(H_0^3(t) \quad H_1^3(t) \quad H_2^3(t) \quad H_3^3(t))$$

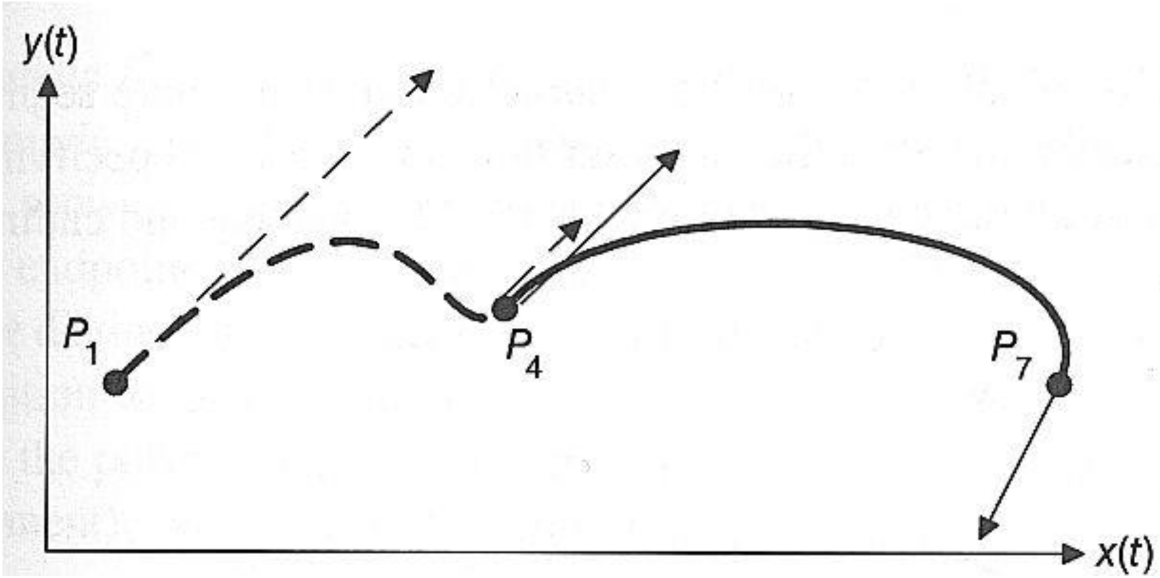
Properties of Hermite Basis Functions

- H_0^3 (H_3^3) interpolates smoothly from 1 to 0
- H_0^3 and H_3^3 have zero derivative at $t = 0$ and $t = 1$
  - No contribution to derivative (H_1^3, H_2^3)
- $H_1^3$and $H_2^3$are zero at $t = 0$ and $t = 1$
  - No contribution to position (H_0^3, H_3^3)
- H_1^3 (H_2^3) has slope 1 at $t = 0$ ($t = 1$)
  - Unit factor for specified derivative vector



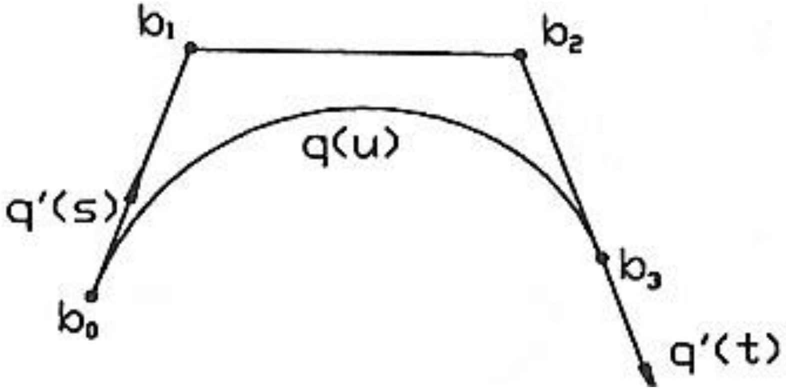

y(t)
Tangent vector direction R_1 at point P_1; magnitude varies for each curve
Tangent vector direction R_4 at point P_4; magnitude fixed for each curve
P_1
P_4
x(t)


y(t)
x(t)


y(t)
P_1
P_4
P_7
x(t)

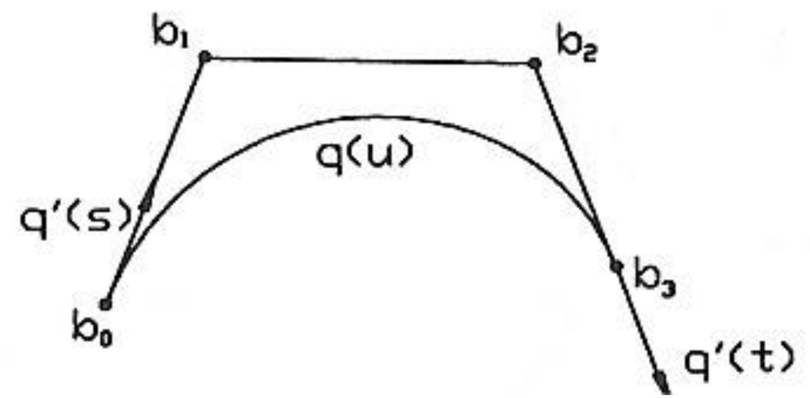
Bézier splines

- Defined by 4 points:
  - b_0, b_3: start and end points
  - b_1, b_2: control points that are approximated
- Searching for $P(t)$ such that:
  - $P(0) = b_0$
  - $P'(0) = 3(b_1 - b_0)$
  - $P'(1) = 3(b_3 - b_2)$
  - $P(1) = b_3$
  - Degree of P is 3



Bézier splines

- Defined by 4 points:
  - b_0, b_3: start and end points
  - b_1, b_2: control points that are approximated
- Searching for $P(t)$ such that:
  - $P(0) = b_0$
  - $P'(0) = 3(b_1 - b_0)$
  - $P'(1) = 3(b_3 - b_2)$
  - $P(1) = b_3$
  - Degree of P is 3



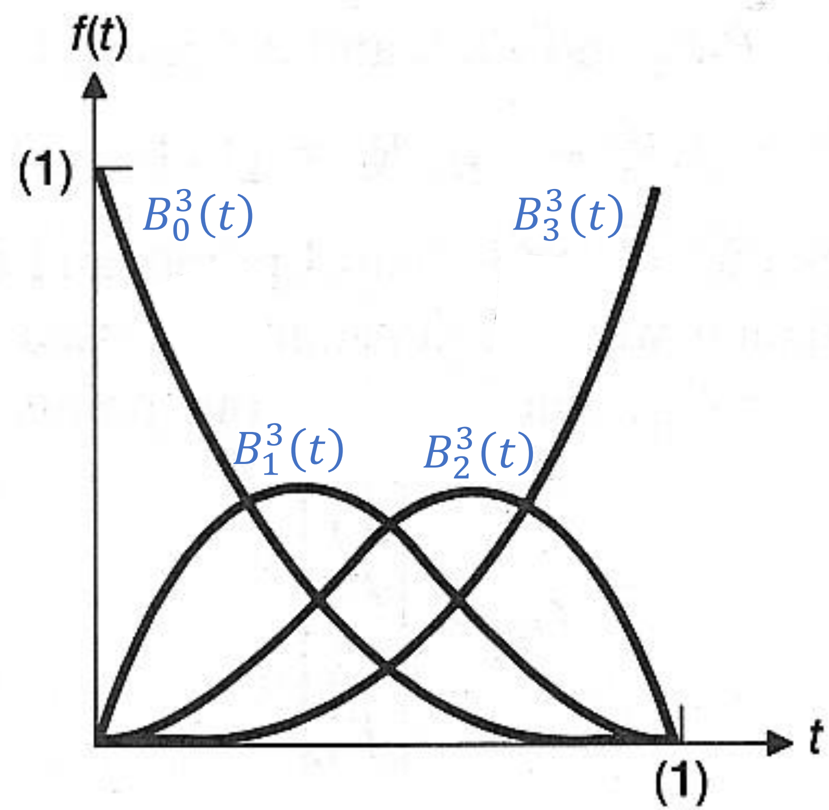
$$\begin{pmatrix} p_1^\top \\ t_1^\top \\ t_2^\top \\ p_2^\top \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ -3 & 3 & 0 & 0 \\ 0 & 0 & -3 & 3 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} b_0^\top \\ b_1^\top \\ b_2^\top \\ b_3^\top \end{pmatrix}$$

$$P(t)^\top = M \cdot H \cdot T_{BH} \cdot G$$

Bézier splines

- Defined by 4 points:
  - b_0, b_3: start and end points
  - b_1, b_2: control points that are approximated
- Searching for $P(t)$ such that:
  - $P(0) = b_0$
  - $P'(0) = 3(b_1 - b_0)$
  - $P'(1) = 3(b_3 - b_2)$
  - $P(1) = b_3$
  - Degree of P is 3
- Basis:
  - $B_0^3(t) = (1-t)^3$
  - $B_1^3(t) = 3t(1-t)^2$
  - $B_2^3(t) = 3t^2(1-t)$
  - $B_3^3(t) = t^3$
- Bernstein polynomial:
  - $B_i^n(t) = \binom{n}{i} t^i (1-t)^{n-i}$

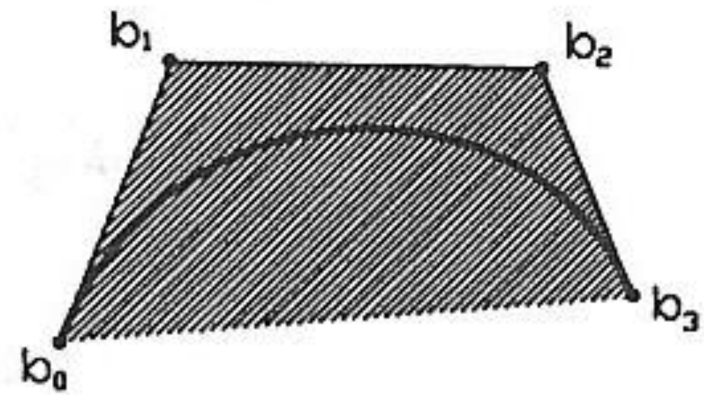
$$B = H \cdot T_{BH} = \begin{pmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 3 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix}$$

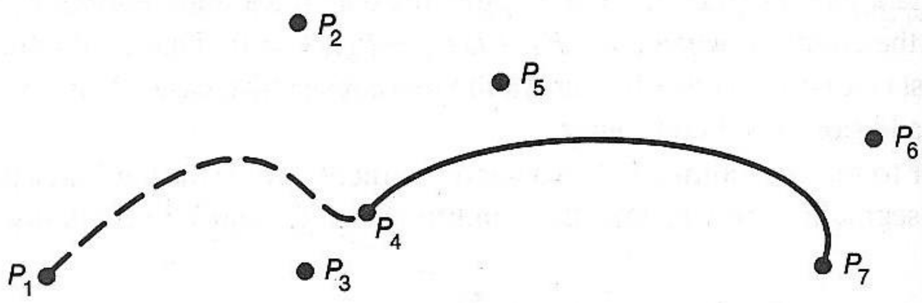


$$P(t) = b_0 B_0^3(t) + b_1 B_1^3(t) + b_2 B_2^3(t) + b_3 B_3^3(t)$$

Advantages:

- End point interpolation
- Tangents explicitly specified
- Smooth joints are simple
  - P_3, P_4, P_5 collinear → G^1 continuous
  - $P_5 - P_4 = P_4 - P_3$ → C^1 continuous
- Geometric meaning of control points
- Affine invariance
- Convex hull property
  - For $0 < t < 1$: $B_i(t) \geq 0$
- Symmetry: $B_i(t) = B_{n-i}(1-t)$





Disadvantages

- Smooth joints need to be maintained explicitly
  - Automatic in B-Splines (and NURBS)

Direct evaluation of the basis functions $P(t) = \sum_i b_i B_i^n(t)$

- Simple but expensive

Use recursion

- Recursive definition of the basis functions

$$B_i^n(t) = \binom{n}{i} t^i (1-t)^{n-i} = tB_{i-1}^{n-1}(t) + (1-t)B_i^{n-1}(t)$$

- Inserting this once yields:

$$P(t) = \sum_{i=0}^{n} b_i^0 B_i^n(t) = \sum_{i=0}^{n-1} b_i^1 B_i^{n-1}(t)$$

- With the new Bézier points given by the recursion

$$b_i^0(t) = b_i$$

$$b_i^k(t) = tb_{i+1}^{k-1}(t) + (1-t)b_i^{k-1}(t)$$

DeCasteljau Algorithm:

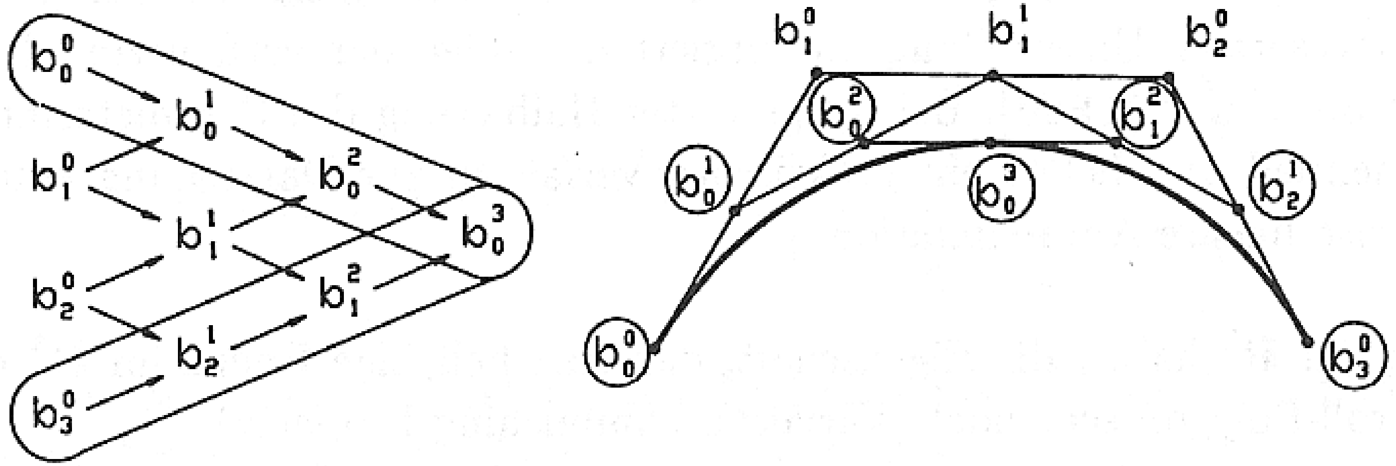
- Recursive degree reduction of the Bezier curve by using the recursion formula for the Bernstein polynomials

$$P(t) = \sum_{i=0}^{n} b_i^0 B_i^n(t) = \sum_{i=0}^{n-1} b_i^1 B_i^{n-1}(t) = \cdots = b_i^n(t) \cdot 1$$

$$b_i^k(t) = t b_{i+1}^{k-1}(t) + (1-t) b_i^{k-1}(t)$$

Example:

- $t = 0.5$

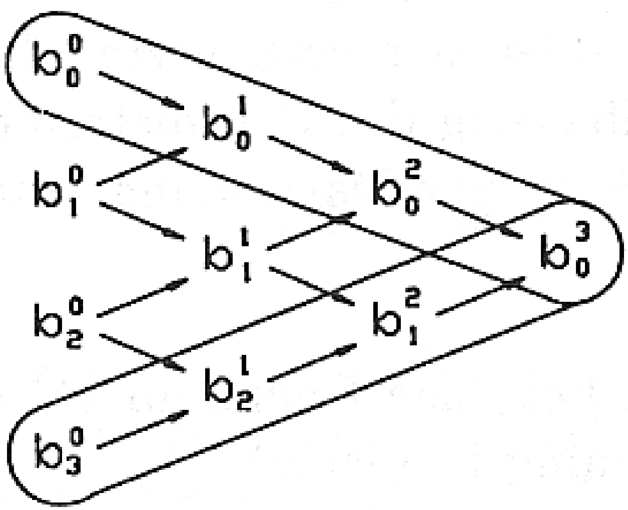


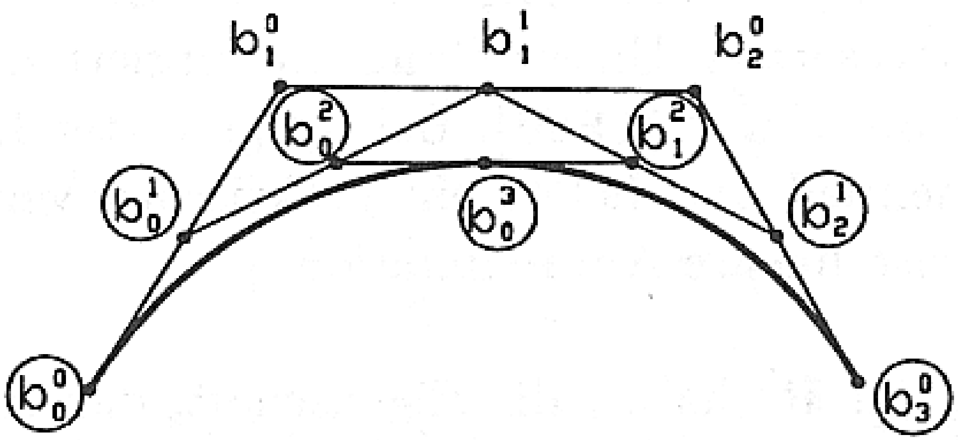
Subdivision using the deCasteljau Algorithm

- Take boundaries of the deCasteljau triangle as new control points for left / right portion of the curve

Extrapolation

- Backwards subdivision
  - Reconstruct triangle from one side



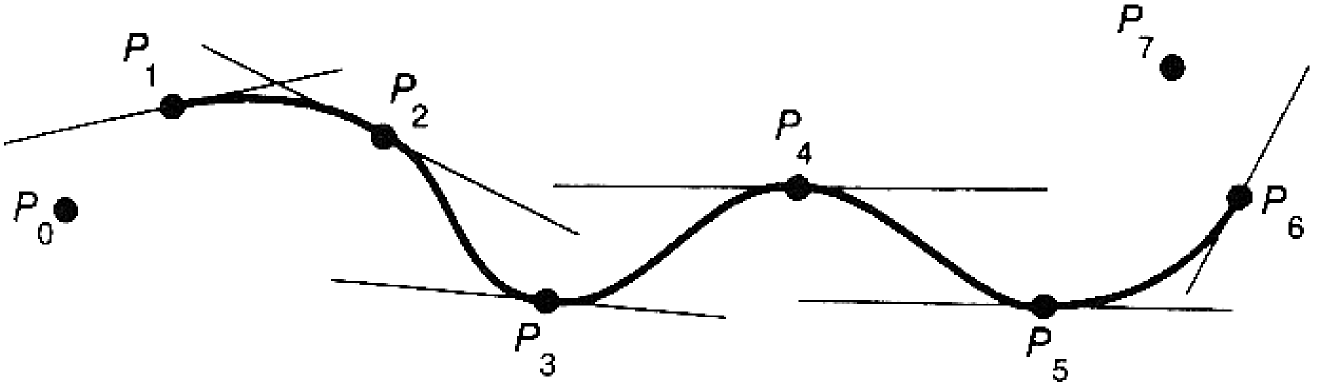


Goal

- Smooth (C^1)-joints between (cubic) spline segments

Algorithm

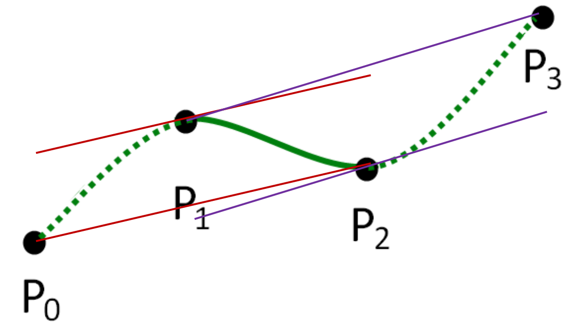
- Tangents given by neighboring points Pi-1 Pi+1
- Construct (cubic) Hermite segments



Advantage

- Arbitrary number of control points
- Interpolation without overshooting
- Local control

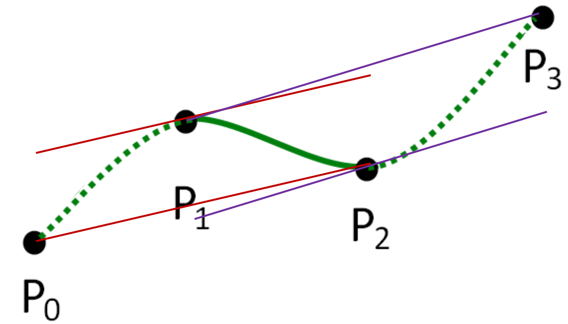
Catmull-Rom splines



- Defined by 4 points:
  - c_1, c_2: start and end points
  - c_0, c_3: neighbor segment points
- Searching for $P(t)$ such that:
  - $P(0) = c_1$
  - $P'(0) = \frac{1}{2}(c_2 - c_0)$
  - $P'(1) = \frac{1}{2}(c_3 - c_1)$
  - $P(1) = c_2$
  - Degree of P is 3

Catmull-Rom splines

- Defined by 4 points:
  - c_1, c_2: start and end points
  - c_0, c_3: neighbor segment points
- Searching for $P(t)$ such that:
  - $P(0) = c_1$
  - $P'(0) = \frac{1}{2}(c_2 - c_0)$
  - $P'(1) = \frac{1}{2}(c_3 - c_1)$
  - $P(1) = c_2$
  - Degree of P is 3

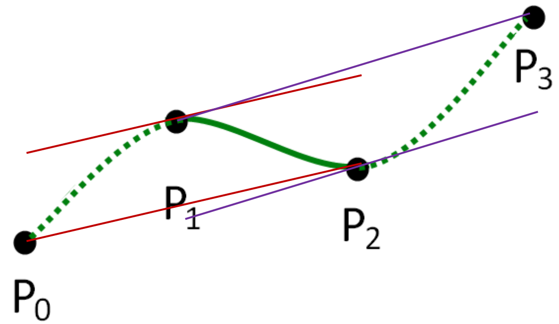


$$\begin{pmatrix} p_1^\intercal \\ t_1^\intercal \\ t_2^\intercal \\ p_2^\intercal \end{pmatrix} = \begin{pmatrix} 0 & 1 & 0 & 0 \\ -0.5 & 0 & 0.5 & 0 \\ 0 & -0.5 & 0 & 0.5 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} c_0^\intercal \\ c_1^\intercal \\ c_2^\intercal \\ c_3^\intercal \end{pmatrix}$$

$$P(t)^\intercal = M \cdot H \cdot T_{CH} \cdot G$$

Catmull-Rom splines

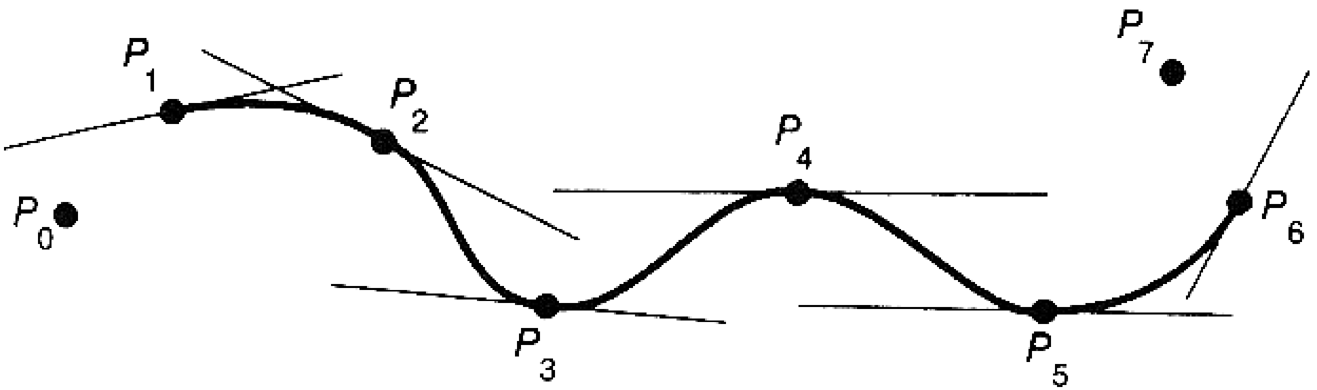
- Defined by 4 points:
  - c_1, c_2: start and end points
  - c_0, c_3: neighbor segment points
- Searching for $P(t)$ such that:
  - $P(0) = c_1$
  - $P'(0) = \frac{1}{2}(c_2 - c_0)$
  - $P'(1) = \frac{1}{2}(c_3 - c_1)$
  - $P(1) = c_2$
  - Degree of P is 3
- Basis:
  - $C_0^3(t) = \frac{1}{2}t(1-t)^2$
  - $C_1^3(t) = \frac{1}{2}(t-1)(3t^2 - 2t - 2)$
  - $C_2^3(t) = -\frac{1}{2}t(3t^2 - 4t - 1)$
  - $C_3^3(t) = \frac{1}{2}t^2(t-1)$



$$C = H \cdot T_{CH} = \frac{1}{2}\begin{pmatrix} -1 & 3 & -3 & 1 \\ 2 & -5 & 4 & -1 \\ -1 & 0 & 1 & 0 \\ 0 & 2 & 0 & 0 \end{pmatrix}$$

Catmull-Rom-Spline

- Piecewise polynomial curve
- Four control points per segment
- For n control points we obtain $(n-3)$ polynomial segments



Application

- Smooth interpolation of a given sequence of points
- Key frame animation, camera movement, *etc.*
- Only G^1-continuity
- Control points should be equidistant in time

Problem

- Often only the control points are given
- How to obtain a suitable parameterization t_i?

Example: *Chord-Length Parameterization*

$$t_0 = 0$$

$$t_i = \sum_{j=1}^{i} dist(P_i - P_{i-1})$$

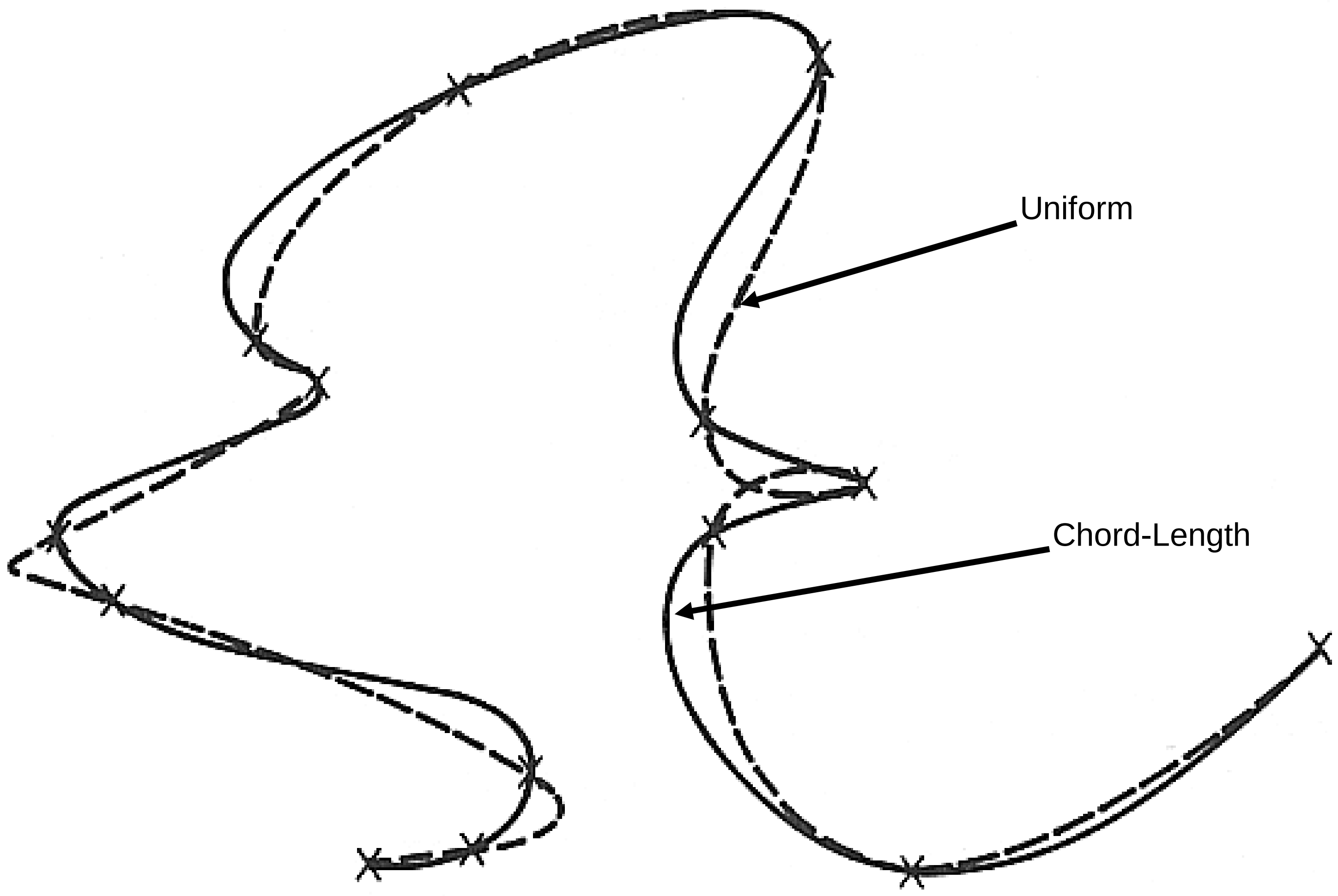
- Arbitrary up to a constant factor

Warning

- Distances are not affine invariant !
- Shape of curves changes under transformations !!

Chord-Length versus uniform Parameterization

- Analog: Think $P(t)$ as a moving object with mass that may overshoot



Goal

- Spline curve with local control and high continuity

Given

- Degree: n
- Control points: $p_0, \ldots, p_m$ (Control polygon, $m \geq n+1$)
- Knots: $t_0, \ldots, t_{m+n+1}$ (Knot vector, weakly monotonic)
- The knot vector defines the parametric locations where segments join

B-Spline Curve

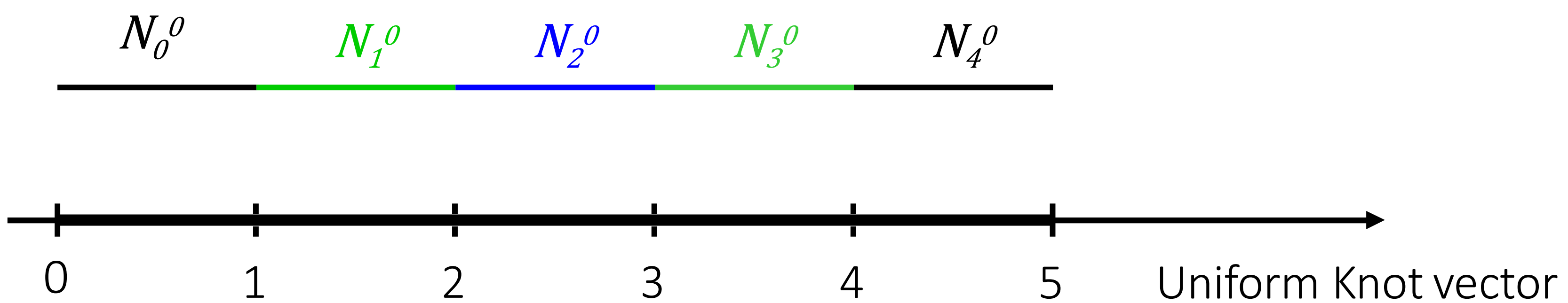
$$P(t) = \sum_{i=0}^{m} N_i^n(t) p_i$$

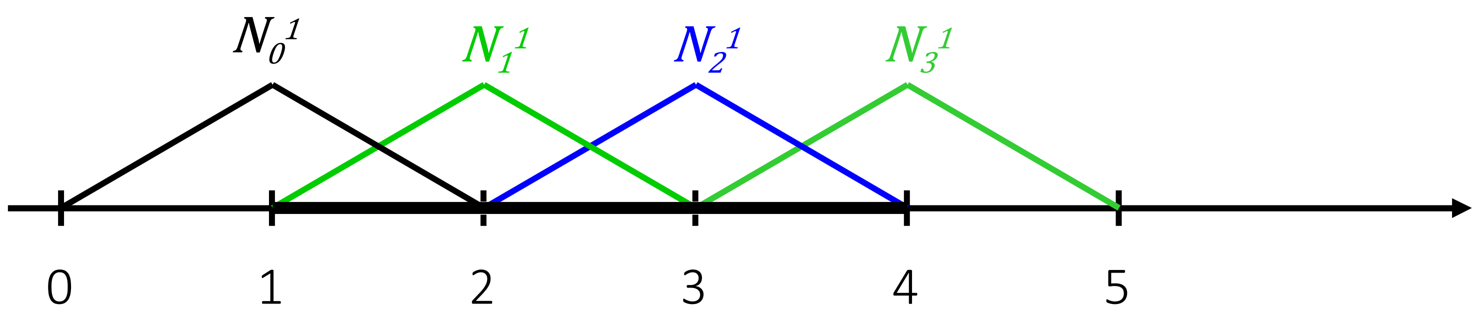
- Continuity:
  - C^{n-1} at simple knots
  - C^{n-k} at knot with multiplicity k

Recursive Definition

$$N_i^0(t) = \begin{cases} 1 & \text{if } t_i < t < t_{i+1} \\ 0 & \text{otherwise} \end{cases}$$

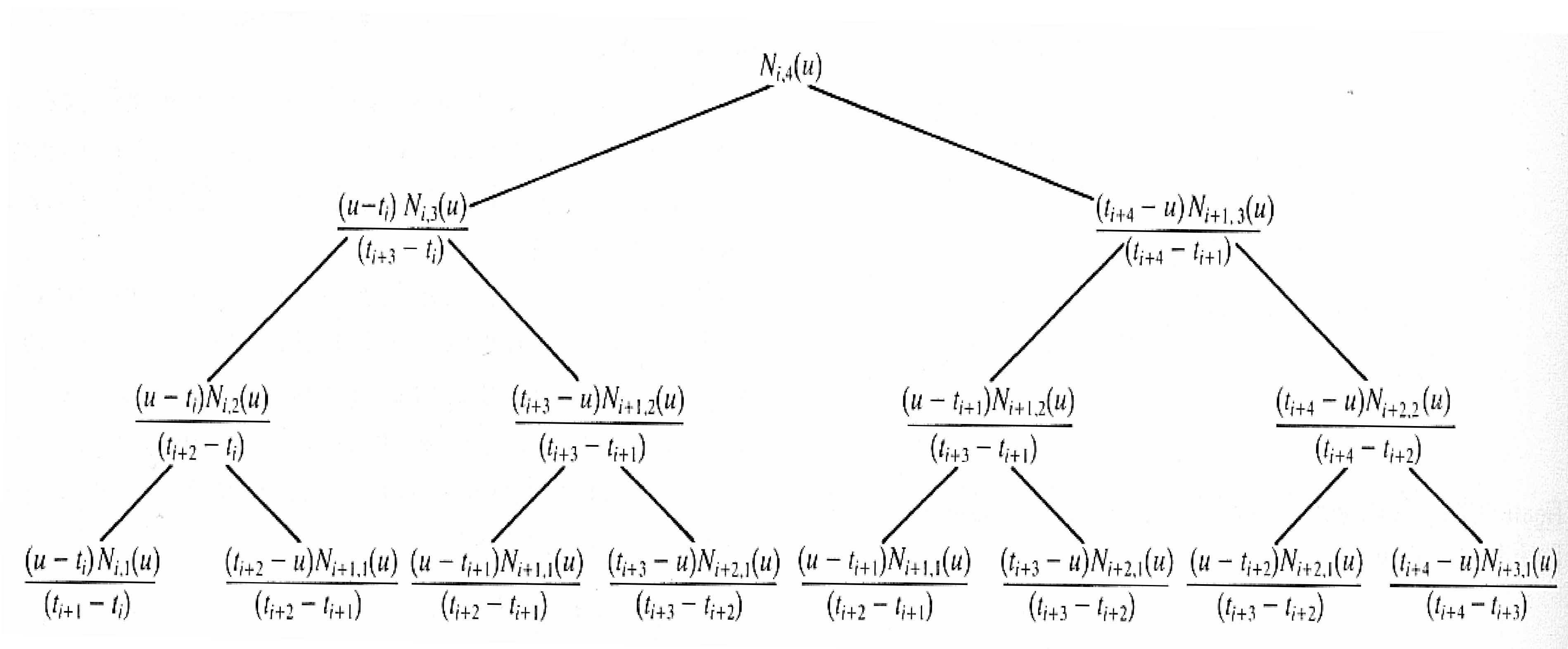
$$N_i^n(t) = \frac{t - t_i}{t_{i+n} - t_i} N_i^{n-1}(t) - \frac{t - t_{i+n+1}}{t_{i+n+1} - t_{i+1}} N_{i+1}^{n-1}(t)$$





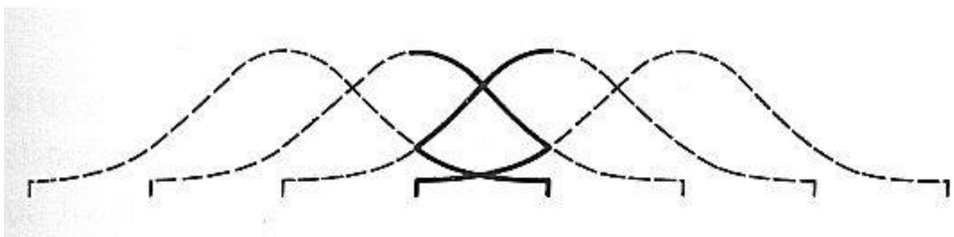
Recursive Definition

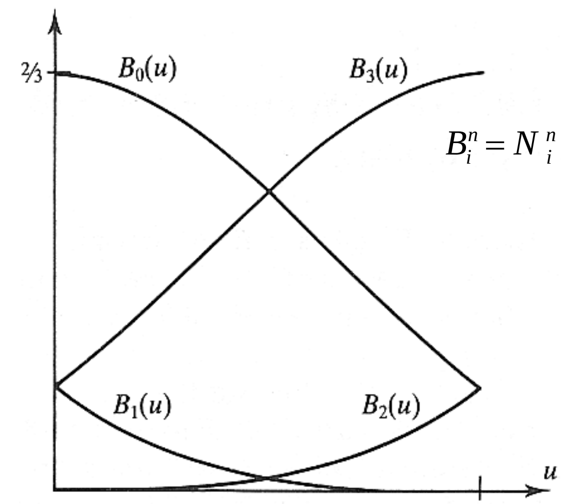
- Degree increases in every step
- Support increases by one knot interval



Uniform Knot Vector

- All knots at integer locations
  - UBS: Uniform B-Spline
- Example: cubic B-Splines



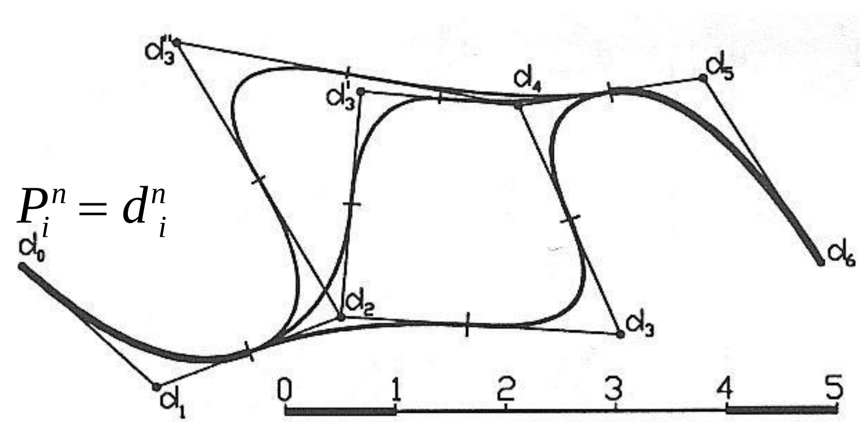


$B_i^n = N_i^n$

Local Support = Localized Changes

- Basis functions affect only $(n+1)$ Spline segments
- Changes are localized

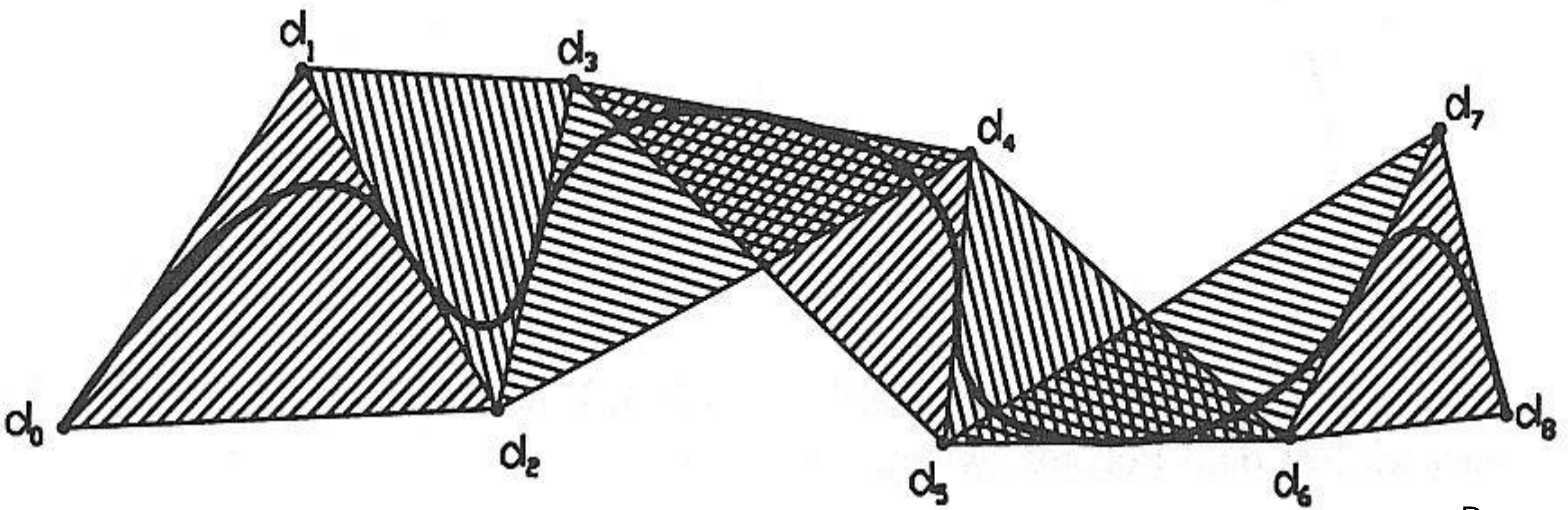
$P_i^n = d_i^n$



Degree 2

Convex Hull Property

- Spline segment lies in convex Hull of (n+1) control points

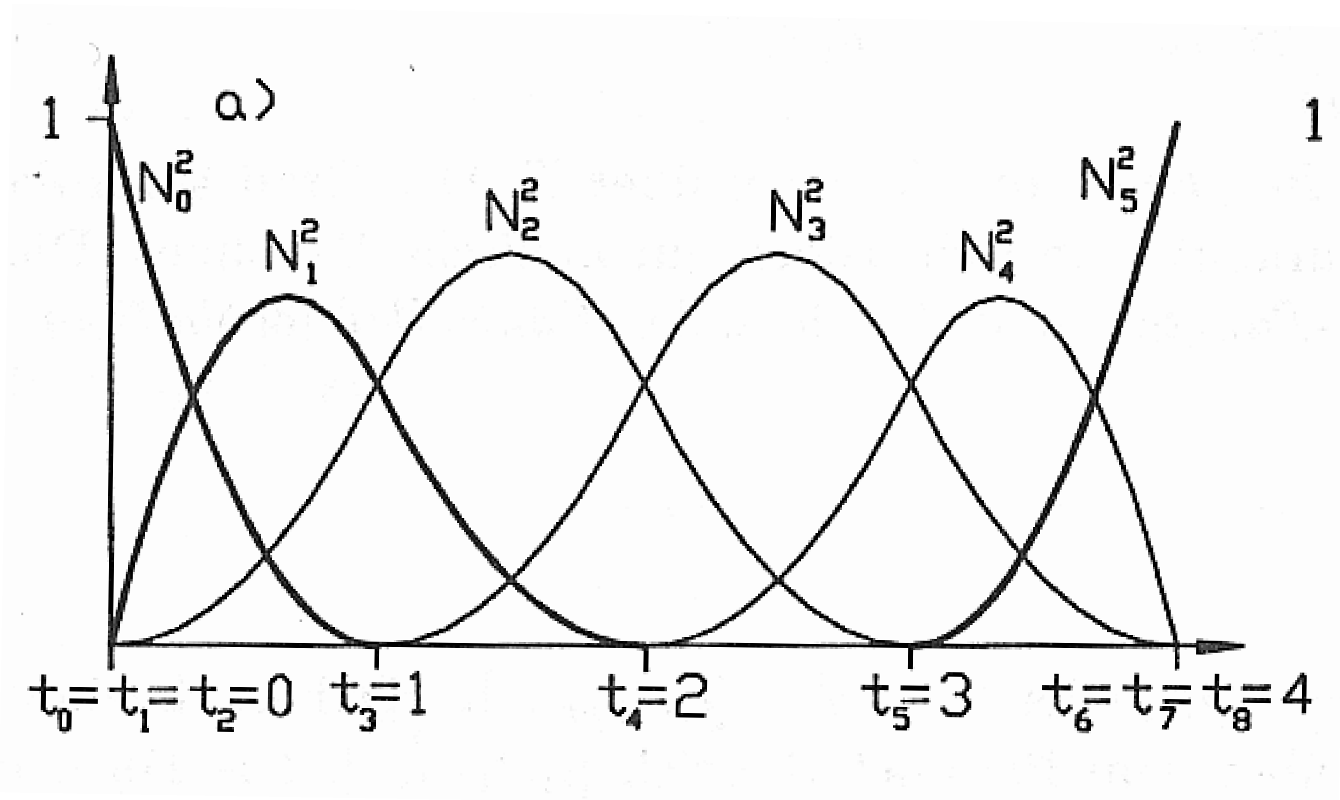


Degree 2

- $(n+1)$ control points lie on a straight line $\Longrightarrow$ curve touches this line
- n control points coincide $\Longrightarrow$ curve interpolates this point and is tangential to the control polygon

Basis Functions on an Interval

- Knots at beginning and end with multiplicity
    - NUBS: Non-uniform B-Splines
- Interpolation of end points and tangents there
- Conversion to Bézier segments via knot insertion



Recursive Definition of Control Points

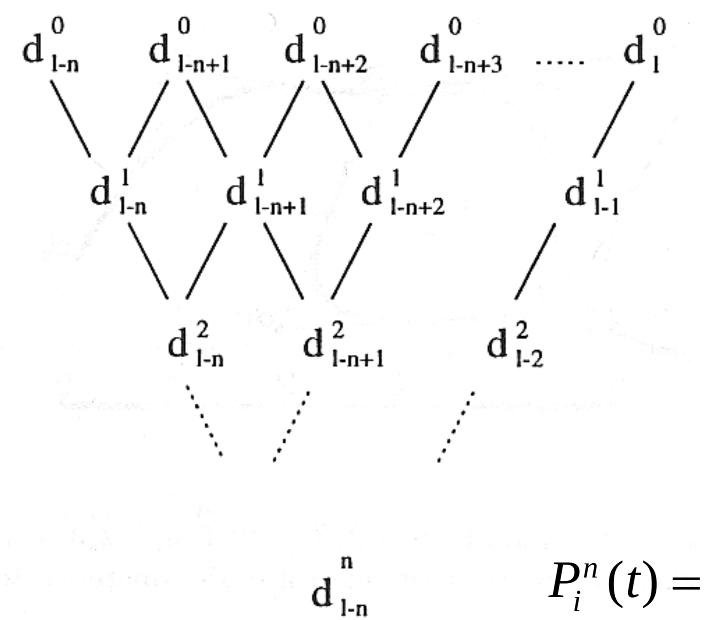
- Evaluation at t: $t_i < t < t_{i+1}$: $i \in \{l-n, \dots, l\}$
- Due to local support only affected by $(n+1)$ control points

$$P_i^r(t) = \left(1 - \frac{t - t_{i+r}}{t_{i+n+1} - t_{i+r}}\right) P_i^{r-1}(t) + \frac{t - t_{i+r}}{t_{i+n+1} - t_{i+r}} P_{i+1}^{r-1}(t)$$

$$P_i^0(t) = P_i$$

Properties

- Affine invariance
- Stable numerical evaluation
  - All coefficients > 0



$$P_i^n(t) = d_i^n$$

Algorithm similar to deBoor

- Given a new knot t
  - $t_i < t < t_{i+1}: i \in \{l-n, \dots, l\}$
- $T^* = T \cup \{t\}$
- New representation of the same curve over T^*

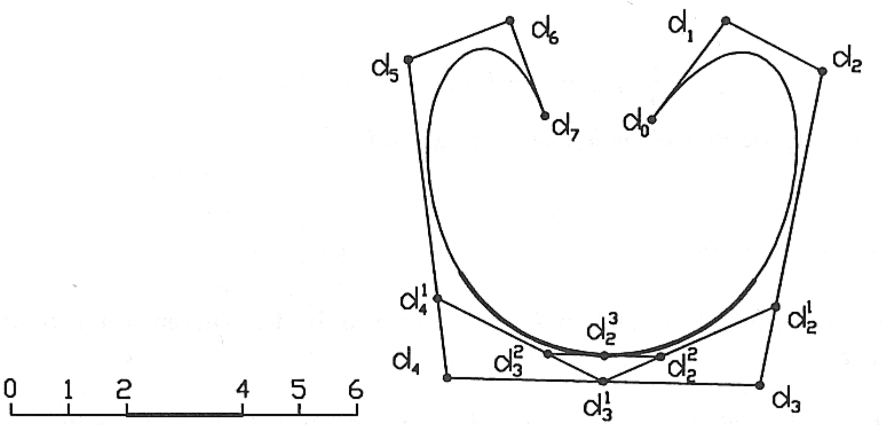
$$P^*(t) = \sum_{i=0}^{m+1} N_i^n(t) P_i^*$$

$$P_i^* = (1 - a_i) P_{i-1} + a_i P_i$$

$$a_i = \begin{cases} 1 & i \le l - n \\ \dfrac{t - t_i}{t_{i+n} - t_i} & l - n + 1 \le i \le l \\ 0 & i \ge l + 1 \end{cases}$$

Applications

- Refinement of curve, display



Consecutive insertion of three knots at $t = 3$ into a cubic B-Spline.
First and last knot have multiplicity n
$T = (0,0,0,0,1,2,4,5,6,6,6,6), l = 5$

B-Spline to Bezier Representation

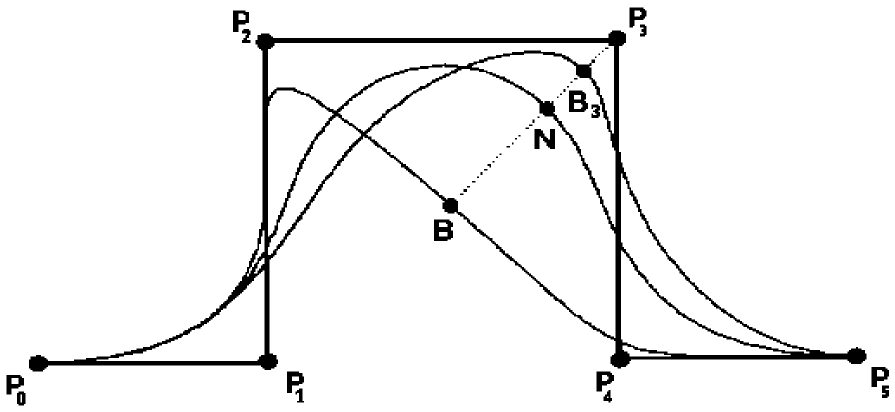
- Remember:
  - Curve interpolates point and is tangential at knots of multiplicity n
- In more detail: If two consecutive knots have multiplicity n
  - The corresponding spline segment is in Bézier from
  - The $(n+1)$ corresponding control polygon form the Bézier control points

Non-Uniform Rational B-Splines

- Homogeneous control points: now with weight w_i
  - $P'_i = (w_i x_i, w_i y_i, w_i z_i, w_i) = w_i P_i$

$$P'(t) = \sum_{i=0}^{m} N_i^n(t) P'_i$$

$$P = \frac{\sum_{i=0}^{m} N_i^n(t) P_i w_i}{\sum_{i=0}^{m} N_i^n(t)\, w_i} = \sum_{i=0}^{m} R_i^n(t) P_i w_i\,, \qquad with\ R_i^n(t) = \frac{N_i^n(t)\, w_i}{\sum_{i=0}^{m} N_i^n(t)\, w_i}$$



Properties

- Piecewise rational functions
- Weights
  - High (relative) weight attract curve towards the point
  - Low weights repel curve from a point
  - Negative weights should be avoided (may introduce singularity)
- Invariant under projective transformations
- Variation-Diminishing-Property (in functional setting)
  - Curve cuts a straight line no more than the control polygon does

Examples: Cubic B-Splines